Veritas InfoScale™ 7.4.1 Virtualization Guide - Linux

VERITAS™

Last updated: 2019-02-01

Legal Notice

Veritas Technologies LLC
500 E Middlefield Road
Mountain View, CA 94043

http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policies. For information about our support offerings and how to contact Technical Support, visit our website:

https://www.veritas.com/support

You can manage your Veritas account information at the following URL:

https://my.veritas.com

If you have questions regarding an existing support agreement, please email the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The latest documentation is available on the Veritas website:

https://sort.veritas.com/documents

Documentation feedback

Your feedback is important to us. Suggest improvements or report errors or omissions to the documentation. Include the document title, document version, chapter title, and section title of the text on which you are reporting. Send feedback to:

infoscaledocs@veritas.com

You can also see documentation information or ask a question on the Veritas community site:

http://www.veritas.com/community/

Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:

https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Section 1

Overview of Veritas InfoScale Solutions used in Linux virtualization

Chapter 1

Overview of supported products and technologies

This chapter includes the following topics:

Overview of the Veritas InfoScale Products Virtualization Guide

This document provides information about Veritas InfoScale products support for Linux virtualization technologies. It contains:

- High-level conceptual information for Veritas InfoScale products and how they function in Linux virtual environments.
- High level implementation information for setting upVeritas InfoScale products in Linux virtual environments.
- Use case chapters with examples of how Veritas InfoScale products can improve performance outcomes for common Linux virtualization use cases.

See “Linux virtualization documentation” on page 212.

About Veritas InfoScale Solutions support for Linux virtualization environments

Veritas InfoScale Solutions products support the following virtualization technologies in Linux environments:

- Kernel-based Virtual Machine (KVM) technology for Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES)
- Red Hat Enterprise Virtualization (RHEV) environment
- Oracle Virtual Machine (OVM) environment
- Microsoft Hyper-V environment
- Linux guests in VMware ESXi environments

Table 1-1 Supported Linux virtualization technologies

Components in Veritas InfoScale Solutions	KVM	RHEV	OVM	Microsoft Hyper-V	Linux in VMware ESXi
Dynamic Multi-Pathing (DMP)	Y	Y	N	Virtual machine only	Y
Storage Foundation (SF)	Y	Y	N	Virtual machine only	Virtual machine only
Cluster Server (VCS)	Y	Y	Y	Virtual machine only	Virtual machine only
Storage Foundation and High Availability (SFHA)	Y	Y	N	Virtual machine only	Virtual machine only
Storage Foundation Cluster File System High Availability (SFCFSHA)	Y	Y	N	Virtual machine only	Virtual machine only
Replicator Option	Virtual machine only	Y	N	Virtual machine only	Virtual machine only

For configuring Veritas InfoScale Solutions in VMware guest environments, see the *Veritas InfoScale™ Solutions Virtualization Guide for ESXi*.

For configuring DMP in VMware environments, see the *Dynamic Multi-Pathing Administrator's Guide for ESXi*.

About SmartIO in the Linux virtualized environment

In the Linux virtualized environment, when you install Veritas InfoScale Solutions in the guest, you can use SmartIO to cache data onto an SSD or any other supported fast device.

SmartIO caching does not support live migration of the guest in KVM and RHEV environments.

For VMware, SmartIO does support vMotion if DMP for VMware (SmartPool) is enabled in the ESXi hypervisor.

See "About the SmartPool feature" on page 16.

Storage Foundation for Oracle RAC is not supported in the Linux virtualized environment.

The following tables show how SmartIO can be used in the Linux virtualized environments.

Table 1-2 shows how SmartIO can be used in the KVM environment.

Table 1-2 Linux: SmartIO support in KVM

Configuration in guest:	Configuration in host:	Caching takes place:	VxVM read caching	VxFS read caching	VxFS writeback caching
SF	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
SFHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
SFCFSHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
Any	SF	in the host	Yes	Yes	Yes
Any	SFCFSHA	in the host	Yes	Yes	Yes

Table 1-3 shows how SmartIO can be used in the RHEV environment.

Table 1-3 Linux: SmartIO support in RHEV

Configuration in guest:	Configuration in host:	Caching takes place:	VxVM read caching	VxFS read caching	VxFS writeback caching
SF	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
SFHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
SFCFSHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
Any	SF	in the host	Yes	Yes	Yes
Any	SFCFSHA	in the host	Yes	Yes	Yes

Table 1-4 shows how SmartIO can be used in the VMware environment.

Table 1-4 Linux: SmartIO support in VMware

Configuration in guest:	Configuration in host:	Caching takes place:	VxVM read caching	VxFS read caching	VxFS writeback caching
SF	DMP for VMware (Optional)	in the guest	Yes	Yes	No
SFHA	DMP for VMware (Optional)	in the guest	Yes	Yes	No
SFCFSHA	DMP for VMware (Optional)	in the guest	Yes	Yes	No

For more information about configuring Veritas InfoScale Solutions in the Linux Virtualization environment, see the *Veritas InfoScale™ Solutions Virtualization Guide for Linux*.

About the SmartPool feature

Dynamic Multi-Pathing for VMware has an operating mode which enables the pooling of locally attached devices such as SSDs at the ESXi host layer. The aggregation of the local devices is called SmartPool. From the SmartPool, you can provision SmartDisks to be used as caching areas by SmartIO in the ESXi guests running Veritas InfoScale. By dividing the SmartPool into several SmartDisks, you can share the caching storage across multiple virtual machines. Using SmartPools gives you the flexibility to move virtual machines across ESXi hosts while SmartIO caching is in progress. Although each host has its own SSD, you can configure

each host to have a comparable view of the SmartDisk. When you use vMotion to migrate the virtual machines that have Veritas InfoScale running, SmartIO shuts down the cache on the source node and restarts the cache on the target host. SmartIO caching stays online during the migration. You can dynamically resize the SmartPool by adding or removing storage devices to the SmartPool.

You can use this mode regardless of whether you are using DMP for VMware to manage storage multi-pathing in the host.

The SmartPool functionality is enabled by installing DMP for VMware in the ESXi host. For the SmartPool functionality, you do not need to have a separate license for DMP.

To use SmartIO in the ESXi guest, you must install Veritas InfoScale in the ESXi guest.

For more information, see the *Veritas InfoScale Virtualization Guide for VMware ESXi*.

If you plan to use DMP for VMware for multi-pathing in the host, you must have the appropriate license.

About Kernel-based Virtual Machine (KVM) technology

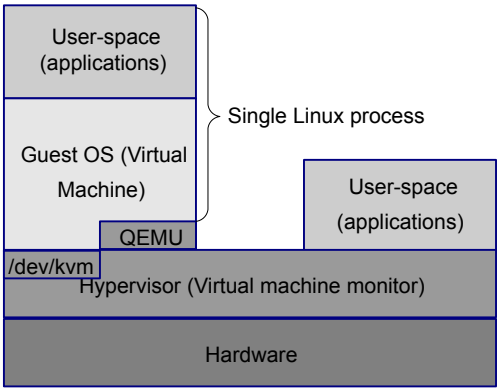
The Veritas InfoScale Solutions can be used in Kernel-based Virtual Machine-based virtualization environments to provide advanced storage management, mission-critical clustering, fail-over, and migration capabilities.

Linux Kernel-based Virtual Machine (KVM) is released by Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES) as a full virtualization solution. KVM differs from other popular alternatives like Xen and VMware in terms of operation, performance and flexibility. KVM comes as a kernel module, with a set of user space utilities to create and manage virtual machines (VM).

Kernel-based Virtual Machine technology includes the following:

- A full virtualization solution for Linux on AMD64 & Intel 64 hardware.
- Each KVM virtualized guest or "VM guest" is run as a single Linux process.
- A hypervisor-independent virtualization API, `libvirt`, which provides a common generic and stable layer to securely manage VM guests on a host.
- A command line tool `virsh` used to manage the VM guests.
- A graphical user interface (GUI) `virt-manager` for managing the VM guests.
- Configuration of each VM guest stored in an XML file.

Figure 1-1 KVM process



This guide illustrates some reference configurations which can be customized to fit most implementations. An assumption is made that the reader understands the Linux operating system, including its architecture, as well as how to configure and manage KVM virtual machines using the management software already provided by Linux. There is also an expectation that the user is familiar with the basic Veritas InfoScale Solutions software and is well versed with its administration and management utilities. Additional details regarding Linux and Veritas InfoScale Solutions software are available in the Additional documentation section.

See "Linux virtualization documentation" on page 212.

Kernel-based Virtual Machine Terminology

Table 1-5 KVM terminology used in this document

Term	Definition
KVM	Kernel-based Virtual Machine
KVMGuest	VCS agent for managing virtual machines in a KVM or RHEV environment.
VM, KVM guest	Virtual machine, also referred to as a KVM virtualized guest.
Host	The physical host on which KVM is installed.
PM	The physical machine running VCS.
VM-VM	VCS-supported configuration in which a cluster is formed between VM guests running inside of the same or different hosts.

Table 1-5 KVM terminology used in this document *(continued)*

Term	Definition
VM-PM	VCS-supported configuration in which a cluster is formed between VM guests and physical machines.
PM-PM	VCS-supported configuration in which a cluster is formed between hosts, and which is mainly used to manage VM guests running inside them.
Bridge	A device bound to a physical network interface on the host which enables any number of VM guests to connect to the local network on the host. It is mapped to a physical NIC which acts as a switch to VM guests.
VirtIO	VirtIO is an abstraction layer for paravirtualized hypervisors in Kernel-based Virtual Machine (VM) technology.

VirtIO disk drives

VirtIO is an abstraction layer for paravirtualized hypervisors in Kernel-based Virtual Machine (VM) technology. Unlike full virtualization, VirtIO requires special paravirtualized drivers running in each VM guest. VirtIO provides support for many devices including network devices and block (disk) devices. Using VirtIO to export block devices to a host allows files, VxVM volumes, DMP meta-nodes, SCSI devices or any other type of block device residing on host to be presented to the VM guest. When SCSI devices are presented to a VM guest using VirtIO, in addition to simple reads and writes, SCSI commands such as SCSI inquiry commands can be performed allowing VxVM in the guest to perform deep device discovery. Running VxVM and DMP in the host and the VM guest provides for consistent naming of SCSI devices from the array, to the host through to the VM guest.

Veritas InfoScale Solutions 7.4.1 supports VirtIO SCSI devices and VirtIO block devices with Linux KVM. virtio-scsi is a new virtual SCSI HBA interface. It is the foundation of an alternative storage implementation for virtual machines, replacing virtio-blk on Red Hat Enterprise Linux (RHEL) with improved scalability and providing standard SCSI command set support.

VirtIO features:

- Dynamically adding devices:
  VirtIO disk devices can be both added and removed from a running VM guest dynamically, without the need of a reboot.

VirtIO limitations:

- Disk caching:

When disks are exported to the VM guest with the cache enabled, the VxVM configuration changes may get cached on the KVM host and not be applied to the disks. When disks are shared between more than one VM guest, such a configuration change is not visble from other VM guest systems than the one which made the change. To avoid potential configuration conflict, caching the host must be disabled (cache=no) while exporting the disks.

- SCSI Commands:
  SCSI devices that are presented as VirtIO devices to a VM guest support a limited subset of the SCSI command set. The KVM hypervisor blocks the restricted commands.
- PGR SCSI-3 Reservations:
  PGR SCSI-3 reservations are not supported on VirtIO block devices. To use SCSI-3 PR operations inside the KVM guest operating system, Veritas recommends that you use virtio-scsi to export SCSI devices to the guest. This limitation is applicable to releases prior to RHEL 6.4.
- DMP Fast Recovery with SCSI devices:
  DMP Fast Recovery bypasses the normal VirtIO read/write mechanism, performing SCSI commands directly against the device. If DMP Fast Recovery is used within the VM guest, caching in the host must be disabled (cache=none), to avoid data integrity issues.
- Thin Reclamation:
  Thin reclamation is not supported on VirtIO devices. The 'WRITE-SAME' command is blocked by the hypervisor. This limitation may be removed in future releases of Linux.
- Resizing devices:
  Linux does not support online disk resizing of VirtIO devices. To re-size a VirtIO device the VM guest must be fully shut down and re-started. Support for online re-sizing of block devices is under evaluation for Linux.
- Maximum number of devices:
  virtio-blk currently has a per-guest limitation of 32 devices. This device limitation includes all VirtIO devices, such as network interfaces and block devices. The device limitation is a result of the current VirtIO implementation where each device acts as a separate PCI device. virtio-scsi solves this limitation by multiplexing numerous storage devices on a single controller. Each device on a virtio-scsi controller is represented as a logical unit, or LUN. The LUNs are grouped into targets. The device limit per target is much larger; each device can have a maximum of 256 targets per controller and 16,384 logical units per target. You can use virtio-scsi instead of virtio-blk to use more than 32(28) disk devices inside the KVM guest.
- VxFS:

In a KVM environment under heavy I/O load, data corruption may occur on VxFS file systems created on LUNs attached as VirtIO block devices. Please refer Red Hat Support Case #00945974 for more details:
https://access.redhat.com/support/cases/00945974

About the RHEV environment

Red Hat Enterprise Virtualization consists of the following components:

- Red Hat Enterprise Virtualization Hypervisor:
  This is a thin hypervisor layer, which is based on Kernel-based Virtual Machine (KVM). As KVM forms a core part of the Linux kernel, it proves to be a very efficient virtualization option.
- Agents and tools:
  These include bundled as well as application-specific agents, and Virtual Desktop Server Manager (VDSM) that runs in the hypervisor. Together, the agents and tools help you administer the virtual machines and the related network and storage.
- Red Hat Enterprise Virtualization platform management infrastructure:
  This provides the interface to view and manage all the system components, machines and images. This management infrastructure provides powerful search capabilities, resource management, live migration, and provisioning.

RHEV terminology

Table 1-6 RHEV terminology used in this document

Term	Definition
KVM	Kernel-based Virtual Machine.
KVMGuest	VCS agent for managing virtual machines in a KVM or RHEV environment.
VM	Virtual machine created in a KVM or RHEV environment.
Host	The physical host on which the virtual machine is created or running.
PM	The physical machine running VCS.
PM-PM	VCS-supported configuration in which a cluster is formed between hosts, and which is mainly used to manage VM guests running inside them.
RHEV	Red Hat Enterprise Virtualization.

Table 1-6 RHEV terminology used in this document *(continued)*

Term	Definition
RHEV-M	Red Hat Enterprise Virtualization Manager is a centralized management web interface for managing the RHEV environment.
RHEL-H	Red Hat Enterprise Linux (RHEL) host that runs a complete version of RHEL, and is managed by RHEV-M.
RHEV-H	Red Hat Enterprise Virtualization - Hypervisor is a minimal installation of Red Hat Enterprise Linux, which supports the creation and operaiton of virtual machines.
VDSM	Virtual Desktop Server Manager. The VDSM service is used by RHEV-M to manage the RHEV-H and RHEL hosts.
REST API	Representational state transfer (REST) API.
Datacenter	A datacenter is a logical entity in a RHEV-M that defines the set of physical and logical resources used in a managed virtual environment such as clusters of hosts, virtual machines, storage and networks.
Cluster	This is a cluster in RHEV-M. A cluster is a collection of physical hosts that share the same storage domains and have the same type of CPU.
Storage Domain	This is the storage infrastructure in RHEV for creating and running virtual machines.
Data Domain	A type of storage domain that holds the disk image of all the virtual machines running in the system, operating system images, and data disks.
ISO Domain	This domain stores ISO files (or logical CDs) used to install and boot operating systems and applications for the virtual machines.

For more information on Red Hat Enterprise Virtualization, see Red Hat Enterprise Virtualization documentation.

Virtualization use cases addressed by Veritas InfoScale products

Veritas InfoScale product components support the following virtualization environment use cases:

Table 1-7 Virtualization use cases addressed by Veritas InfoScale Solutions in a Linux environment

Virtualization use case	Recommended Veritas InfoScale products	Virtualization technology supported	Implementation details
Server consolidation	SFHA or SFCFSHA in the guest	Red Hat Enterprise Linux (RHEL) KVM SUSE Linux Enterprise Server (SLES) KVM RHEV Linux on Microsoft Hyper-V	How to run virtual machines as physical servers. See "Server consolidation" on page 108.
Physical to virtual migration	SF in the host SFHA or SFCFSHA	RHEL KVM SLES KVM RHEV	How to migrate data from physical to virtual environments safely and easily. See "Physical to virtual migration" on page 111.
Simplified management	SFHA or SFCFSHA in the host	RHEL KVM SLES KVM RHEV	How to manage virtual machines using the same command set, storage namespace, and environment as in a non-virtual environment. See "Simplified management" on page 118.
Application failover	VCS or SFHA in the guest	RHEL KVM Red Hat Enterprise Virtualization (RHEV) SLES KVM Linux on VMware ESXi Linux on Microsoft Hyper-V	How to manage application monitoring on virtual machines. How to manage application failover on virtual machines. See " Cluster Server In a KVM Environment Architecture Summary" on page 128.

Table 1-7 Virtualization use cases addressed by Veritas InfoScale Solutions in a Linux environment *(continued)*

Virtualization use case	Recommended Veritas InfoScale products	Virtualization technology supported	Implementation details
Virtual-to-virtual (in-guest) clustering	VCS in the guest	RHEL KVM RHEV SLES KVM Linux on Microsoft Hyper-V Linux on VMware ESXi Oracle Virtual Machine (OVM)	How to configure VCS for virtual-to-virtual clustering. See "Installing and configuring Cluster Server for Red Hat Enterprise Virtualization (RHEV) virtual-to-virtual clustering" on page 148. See "Installing and configuring Cluster Server for Red Hat Enterprise Virtualization (RHEV) virtual-to-virtual clustering" on page 148. See " Installing and configuring Cluster Server with Microsoft Hyper-V virtual-to-virtual clustering" on page 151. See "Installing and configuring Cluster Server for Oracle Virtual Machine (OVM) virtual-to-virtual clustering" on page 153.
Virtual machine availability	VCS in the host	RHEL KVM RHEV SLES KVM	How to manage virtual machine failover. See "VCS in host monitoring the Virtual Machine as a resource" on page 135.

Table 1-7 Virtualization use cases addressed by Veritas InfoScale Solutions in a Linux environment *(continued)*

Virtualization use case	Recommended Veritas InfoScale products	Virtualization technology supported	Implementation details
Virtual machine Live Migration	SFCFSHA in the host	RHEL KVM SLES KVM RHEV	How to use features such as instant snapshots to contain boot images and manage them from a central location in the host. How to enable use of SSDs or HDDs by leveraging Flexible Shared Storage (FSS). FSS value proposition: Storage provisioning offered by Veritas InfoScale Solutions in the host that would allow storage to be provisioned to virtual machines from a single pool having the same namespace across machines in a hypervisor cluster. The cluster need not have shared storage as local storage can be shared using the FSS option. See "About live migration" on page 137.
Virtual machine Live Migration	SFCFSHA in the host	RHEV	How to use features such as instant snapshots to contain boot images and manage them from a central location in the host. See "About live migration" on page 137.
Disaster recovery (DR) in the virtual environment	SFHA or SFCFSHA in the host	RHEV	How to configure virtual machines for disaster recovery. How to configure SF as backend storage for virtual machines. How to enable use of SSDs or HDDs by leveraging Flexible Shared Storage (FSS) See "About disaster recovery for Red Hat Enterprise Virtualization virtual machines" on page 156.

Table 1-7 Virtualization use cases addressed by Veritas InfoScale Solutions in a Linux environment *(continued)*

Virtualization use case	Recommended Veritas InfoScale products	Virtualization technology supported	Implementation details
Application to storage visibility	Configuration for Veritas InfoScale Operations Manager use case	RHEL KVM SLES KVM RHEV Linux on VMware ESXi Microsoft Hyper-V	How to configure for storage to application visibility. See "About storage to application visibility using " on page 103.
Managing docker containers	InfoScale Enterprise in the host	RHEV	How to manage storage, ensure high availability , migrate, and recover docker containers. See "About managing Docker containers with InfoScale Enterprise product" on page 172.
Multi-tier Business service support	Veritas InfoScale Operations Manager, Virtual Business Service (VBS)	RHEL KVM SLES KVM RHEV	How to discover and configure devices for multi-tier application. See "About Kernel-based Virtual Machine (KVM) virtualization discovery in Veritas InfoScale Operations Manager" on page 104. See "About Microsoft Hyper-V virtualization discovery" on page 105.

Note: ApplicationHA is supported in the RHEL KVM environment only.

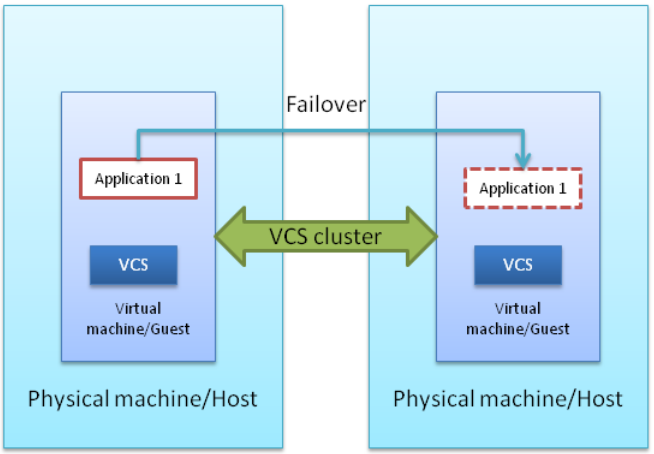
About virtual-to-virtual (in-guest) clustering and failover

When you run Cluster Server (VCS) in multiple guest virtual machines, you can create guest-to-guest (also called virtual-to-virtual) clusters. You can use VCS to monitor individual applications running inside each guest. In case of application

failure, VCS can fail over the application to another guest virtual machine in the virtual-to-virtual cluster.

The following figure illustrates a sample in-guest VCS deployment in one virtual machine each across two physical hosts.

Figure 1-2 VCS in-guest clustering



The virtual machines in the cluster can either be on the same physical host or on different physical hosts. VCS is installed in the virtual machines and creates a cluster. This is just like the cluster that VCS creates among physical systems. The cluster monitors the applications and services that run inside the virtual machines. Any faulted application or service is failed over to another virtual machine in the cluster.

To ensure application failover, application data must reside on storage shared by member virtual machines within the cluster.

Note: In this configuration, since VCS runs inside a virtual machine, VCS cannot fail over the virtual machine itself.

VCS can be deployed inside guest virtual machines (in-guest support) in the following virtualization environments:

- Microsoft Hyper-V
- Red Hat Enterprise Virtualization (RHEV)

- Oracle Virtual Machine (Oracle VM)
- Kernel-based Virtual Machine (KVM) technology for Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES)
- Linux guests in VMware ESXi environments

Section 2

Implementing a basic KVM environment

Chapter 2

Getting started with basic KVM

This chapter includes the following topics:

Creating and launching a kernel-based virtual machine (KVM) host

KVM is available as part of Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES). Management for RHEL KVM is provided through separate RPMs that can be downloaded into the standard RHEL installation. Management for SLES KVM is provided through SLES or through separate RPMs that can be downloaded into the standard SLES installation.

The `virt-manager` tool provides a very simple, easy-to-use and intuitive GUI interface for all virtual machine operations, along with `virt-viewer`. A command

line alternative, `virsh`, also provides a shell that can be used to create and manage virtual machines using a rich set of commands. The features provided by these tools include taking snapshots of virtual machines, creating virtual networks and live migration of virtual machines to another KVM host.

Once you have configured the required hardware setup:

- Install KVM on the target systems.
  See "Linux virtualization documentation" on page 212.
- Create and launch the required KVM virtual machines.
  See "Setting up a KVM guest" on page 31.
- Proceed to install the required SFHA product on the guest or host:
  See "Installing Veritas InfoScale Solutions in the kernel-based virtual machine environment" on page 45.
  See "Installing and configuring Cluster Server in a kernel-based virtual machine (KVM) environment" on page 47.

RHEL-based KVM installation and usage

You can list the available groups for virtualization from all yum repos with using the following `yum` command:

```
# yum grouplist|grep -i virtualization
```

This command lists the package group that has 'virtualization' as a substring in the group name among the list of all group names and does not install the virtualization RPM.

Subsequently, you can install the virtualization RPM with the following command:

```
# yum groupinstall "Virtualization"
```

Setting up a KVM guest

The following is a high-level overview of the steps required for setting up KVM. For detailed instructions, refer to the applicable Linux documentation.

1. Before creating KVM guests, ensure that CPU and memory resources are available to create KVM guests on all nodes in the cluster.
2. Make sure that the required KVM packages are installed on the hosts.
3. Make sure that the service libvirtd is running on the hosts where KVM guests are to be created.

4. Create KVM guests. For network configuration, refer to the *Network configuration for VM-VM cluster* in Appendix A..
5. Install the operating system in the KVM guests.
6. Repeat the above steps for all KVM guests that you want to be a part of the cluster.
7. Install VCS on all the KVM guests. For information about installing VCS, refer to the *Veritas InfoScale Installation Guide.*
8. Configure the VCS resources that you want VCS to manage. For more information, refer to the VCS documentation.

See "Network configuration for VM-VM cluster" on page 59.

About setting up KVM with Veritas InfoScale Solutions

Before setting up your virtual environment, verify that your planned configuration will meet the system requirements, licensing and other considerations for installation with Veritas InfoScale Solutions products.

- Licensing: customers running Storage Foundation (SF) or Storage Foundation Cluster File System High Availability (SFCFSHA) in a kernel-based virtual machine (KVM) environment are entitled to use an unlimited number of guests on each licensed server or CPU.
- Red Hat and SUSE system requirements: see Table 2-1
- Veritas product requirements: see Table 2-2
- *Release Notes*: each Veritas product contains last minute news and important details for each product, including updates to system requirements and supported software. Review the *Release Notes* for the latest information before you start installing the product.
  The product documentation is available on the Web at the following location: https://sort.veritas.com/documents

Table 2-1 Red Hat and SUSE system requirements

	Red Hat Enterprise Linux (RHEL)	**SUSE Linux Enterprise Server (SLES)**
Supported architecture	■ Intel 64 ■ AMD 64	■ Intel 64 ■ AMD 64

Table 2-1 Red Hat and SUSE system requirements *(continued)*

	Red Hat Enterprise Linux (RHEL)	**SUSE Linux Enterprise Server (SLES)**
Minimum system requirement	■ 6 GB free disk space ■ 2 GB of RAM	■ 6 GB free disk space ■ 2 GB of RAM
Recommended system requirement	■ 6 GB plus the required disk space recommended by the guest operating system per guest. For most operating systems more than 6 GB of disk space is recommended ■ One processor core or hyper-thread for each virtualized CPU and one for the host ■ 2 GB of RAM plus additional RAM for virtualized guests	■ 6 GB plus the required disk space recommended by the guest operating system per guest. For most operating systems more than 6 GB of disk space is recommended ■ One processor core or hyper-thread for each virtualized CPU and one for the host ■ 2 GB of RAM plus additional RAM for virtualized guests
Hardware requirement	Full virtualization-enabled CPU	Full virtualization-enabled CPU
Veritas InfoScale Solutions	Veritas InfoScale Solutions 7.4.1	Veritas InfoScale Solutions 7.4.1
Supported OS version in the host	[KVM, RHEV] RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 3, Update 4, Update 5, Update 6 [KVM only] RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 3, Update 4, Update 5, Update 6	SLES11 SP2, SP3 SLES12 SP2, SP3

Table 2-1 Red Hat and SUSE system requirements *(continued)*

	Red Hat Enterprise Linux (RHEL)	SUSE Linux Enterprise Server (SLES)
Supported OS version in the virtual machine	[KVM, RHEV] RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 3, Update 4, Update 5, Update 6 [KVM only] RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 3, Update 4, Update 5, Update 6	SLES11 SP2, SP3 SLES12 SP2, SP3

Table 2-2 Veritas product requirements

Hardware	https://www.veritas.com/support/en_US/article.000126344
Software	■ Dynamic Multi-Pathing Used for storage visibility on KVM hosts and guest virtual machines ■ Storage Foundation Used for storage management on KVM hosts and guest virtual machines ■ Storage Foundation HA Used for storage management and clustering on KVM hosts and guest virtual machines ■ Storage Foundation Cluster File System High Availability 7.4.1 Used for storage management and clustering multiple KVM hosts to enable live migration of guest virtual machines ■ Cluster Server Used for virtual machine monitoring, migration, and failover ■ Veritas Operations Manager 5.0 Used for application visibility and virtual host management
Storage	■ Shared storage for holding the guest image. (VM failover) ■ Shared storage for holding the application data. (Application failover) ■ Local storage with Flexible Shared Storage (FSS) option enabled for VM and application failover
Networking	■ Configure the guest for communication over the public network ■ Setup virtual interfaces for private communication.

Table 2-2 Veritas product requirements *(continued)*

Documentation: see the product release notes to for the most current system requirements, limitations, and known issues:	■ *Veritas InfoScale Release Notes* ■ Veritas Services and Operations Readiness Tools (SORT): https://sort.veritas.com/documents

Table 2-3 VCS system requirements for KVM-supported Red Hat Enterprise Linux configurations

VCS version	7.4.1
Supported OS version in host	RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 4, Update 5, Update 6 SLES11 SP3, SP4 SLES12 SP2, SP3
Supported OS in VM guest	RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 4, Update 5, Update 6 SLES11 SP3, SP4 SLES12 SP2, SP3
Hardware requirement	Full virtualization-enabled CPU

Limitations and unsupported kernel-based virtual machine features

For more information on limitations and known issues, see the *Veritas InfoScale Release Notes* for Linux.

For KVM related limitations, see the Virtualization technology provider (RHEL or SLES) release notes.

Veritas InfoScale Solutions configuration options for the kernel-based virtual machines environment

Veritas InfoScale Solutions products support the configurations listed in Table 2-4. The configurations profiled in the table below are the minimum required to achieve the storage and availability objectives listed. You can mix and match the use of Veritas InfoScale Solutions products as needed to achieve the desired level of storage visibility, management, replication support using VVR, availability, and

cluster failover for your kernel-based virtual machines (KVM) hosts and guest virtual machines.

Table 2-4 Veritas InfoScale Solutions supported configuration options in the KVM environment

Objective	Recommended Veritas InfoScale Solutions product configuration
Storage visibility for KVM guest virtual machines	Dynamic Multi-Pathing (DMP) in the KVM guest virtual machines
Storage visibility for KVM hosts	DMP in the KVM hosts
Storage management features and replication support using VVR for KVM guest virtual machines	Storage Foundation (SF) in the KVM guest virtual machines See " Storage Foundation in the virtualized guest machine" on page 38.
Advanced storage management features and replication support using VVR for KVM hosts	Storage Foundation Cluster File System (SFCFSHA) in the KVM hosts
End-to-end storage visibility in KVM hosts and guest virtual machines	DMP in the KVM host and guest virtual machines
Storage management features and replication support using VVR in the KVM guest virtual machines and storage visibility in in the KVM host	DMP in the KVM host and SF in the KVM guest virtual machines See " Dynamic Multi-Pathing in the KVM host and Storage Foundation HA in the KVM guest virtual machine" on page 41.
Virtual machine monitoring, migration, and failover for KVM hosts	Cluster Server (VCS) in the KVM hosts See " Cluster Server in the KVM host" on page 42.
Application failover for KVM guest virtual machines	VCS in the KVM guest virtual machines See "Cluster Server in the guest" on page 43.
Application availability and virtual machine availability	ApplicationHA in the KVM guest virtual machines and VCS in the KVM host
Application failover across KVM guest virtual machines and physical hosts	VCS in KVM guest virtual machines and KVM physical host machines See "Cluster Server in a cluster across virtual machine guests and physical machines" on page 44.

See "About setting up KVM with Veritas InfoScale Solutions" on page 32.

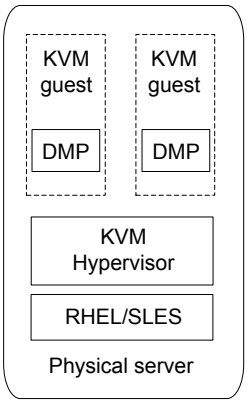
See "Virtualization use cases addressed by Veritas InfoScale products" on page 22.

Dynamic Multi-Pathing in the KVM guest virtualized machine

Use Dynamic Multi-Pathing (DMP) to provide storage visibility in KVM guest virtualized machines. DMP in the KVM guest virtualized machine provides:

- Multi-pathing functionality for the operating system devices configured in the guest
- DMP metadevices (also known as DMP nodes) to represent all the device paths to the same physical LUN
- Support for enclosure-based naming
- Support for standard array types

Figure 2-1 Dynamic Multi-Pathing in the guest



For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

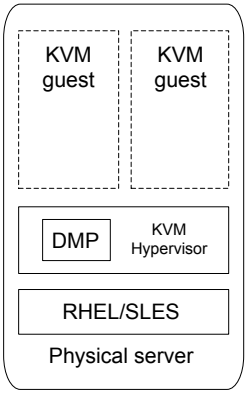
Dynamic Multi-Pathing in the KVM host

Use Dynamic Multi-Pathing (DMP) to provide storage visibility in the KVM hosts. Using DMP in the KVM host enables:

- Centralized multi-pathing functionality
- Enables active/passive array high performance failover
- Centralized storage path management

- Fast proactive failover
- Event notification

Figure 2-2 Dynamic Multi-Pathing in the KVM host



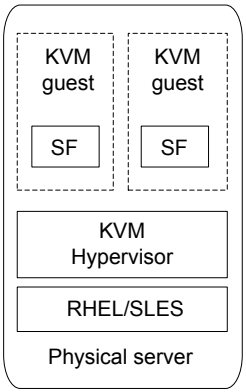
For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

Storage Foundation in the virtualized guest machine

Use Storage Foundation (SF) in the guest to provide storage management functionality for KVM guest virtual machine resources. Storage Foundation enables you to manage KVM guest storage resources more easily by providing:

- Enhanced database performance
- Point-in-time copy features for data back-up, recovery, and processing
- Options for setting policies to optimize storage
- Methods for migrating data easily and reliably
- Replication support

Figure 2-3 Storage Foundation in the virtualized guest machine



For more information on Storage Foundation features, see the *Storage Foundation Administrator's Guide*.

Enabling I/O fencing in KVM guests

InfoScale Enterprise solution supports SCSI-3 PR fencing in the KVM guest environment backed by DMP devices in KVM hosts.

Note: KVM guests and KVM hosts should have InfoScale Storage Foundation or InfoScale Foundation Cluster File System High Availability 7.3.1 version

You can enable I/O fencing in configurations where, single InfoScale instance is configured inside a physical server. Moreover, InfoScale instance can also be configured inside KVM guest or directly in physical server.

See the *Cluster Server Administrator's Guide for information about configuring I/O fencing.*

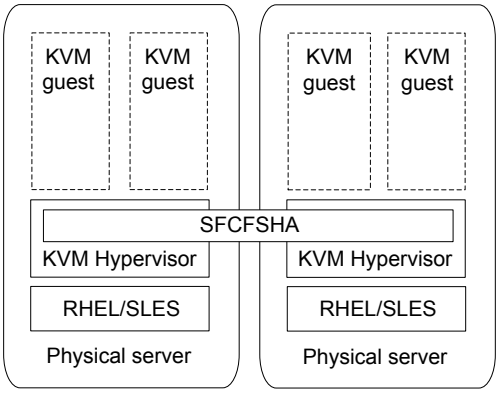
Storage Foundation Cluster File System High Availability in the KVM host

Use Storage Foundation Cluster File System High Availability (SFCFSHA) to provide advanced storage management functionality for the KVM host. SFCFSHA enables you to manage your KVM host storage resources more easily by providing:

- Enhanced database performance
- Point-in-time copy features for data back-up, recovery, and processing
- Options for setting policies to optimize storage

- Methods for migrating data easily and reliably
- Replication support
- High availability for virtual machines
- High availability and disaster recovery for virtual machines
- Simplified management of virtual machines

Figure 2-4 Storage Foundation Cluster File System High Availability in the KVM host



For more information on Storage Foundation features, see the *Storage Foundation™ Cluster File System High Availability Administrator's Guide*.

Dynamic Multi-Pathing in the KVM host and guest virtual machine

Use Dynamic Multi-Pathing (DMP) to provide end-to-end storage visibility across both the KVM host and guest virtual machine. Using DMP in the KVM guest virtualized machine provides:

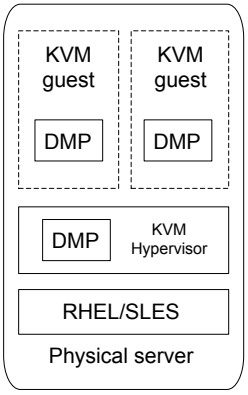
- Multi-pathing functionality for the operating system devices configured in the guest
- DMP metadevices (also known as DMP nodes) to represent all the device paths to the same physical LUN
- Support for enclosure-based naming
- Support for standard array types

Using DMP in the KVM host enables:

- Centralized multi-pathing functionality
- Enables active/passive array high performance failover

- Centralized storage path management
- Fast proactive failover
- Event notification

Figure 2-5 Dynamic Multi-Pathing in the KVM virtualized guest and the KVM host



For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

Dynamic Multi-Pathing in the KVM host and Storage Foundation HA in the KVM guest virtual machine

Use Storage Foundation and High Availability (SFHA) in the guest in combination with Dynamic Multi-Pathing (DMP) in the KVM host to combine storage management functionality for KVM guest virtual machine resources and storage visibility in the KVM host.

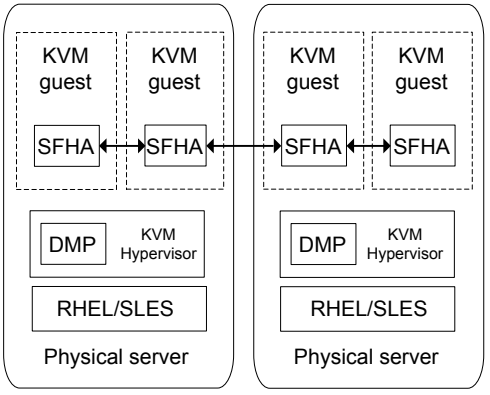
Using SFHA in the KVM guest provides:

- Enhanced database performance
- Point-in-time copy features for data back-up, recovery, and processing
- Options for setting policies to optimize storage
- Methods for migrating data easily and reliably
- Replication support
- High availability for applications running inside virtual machines

Using DMP in the host provides:

- Centralized multi-pathing functionality
- Fast proactive failover.
- Event notification

Figure 2-6 Storage Foundation HA in the KVM guest virtual machine and DMP in the KVM host



For more information on SFHA features, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

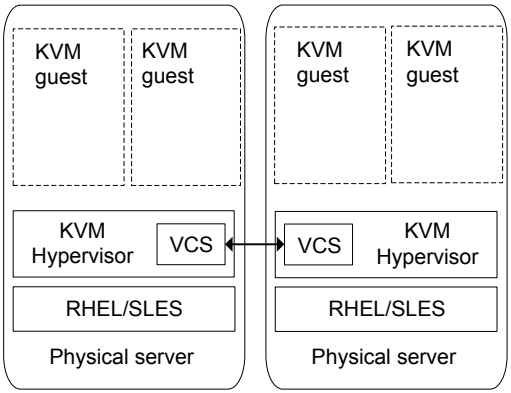
For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

Cluster Server in the KVM host

Use Cluster Server (VCS) to provide virtual machine monitoring and failover to another KVM host. VCS enables the following for KVM hosts:

- Connects multiple, independent systems into a management framework for increased availability.
- Enables nodes to cooperate at the software level to form a cluster.
- Links commodity hardware with intelligent software to provide application failover and control.
- Enables other nodes to take predefined actions when a monitored application fails, for instance to take over and bring up applications elsewhere in the cluster.

Figure 2-7 Cluster Server in the KVM host



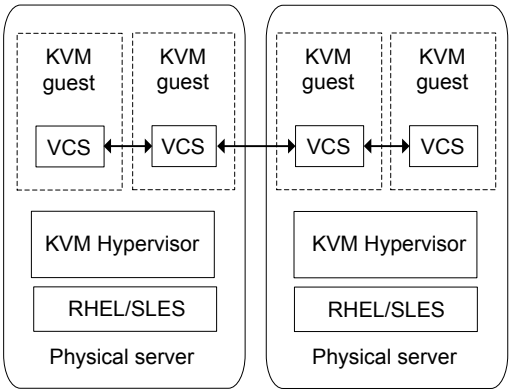
For more information on Cluster Server features, see the *Cluster Server Administrator's Guide*.

Cluster Server in the guest

Use Cluster Server (VCS) to provide application monitoring and failover to another KVM guest.

- Connects multiple, independent systems into a management framework for increased availability
- Enables nodes to cooperate at the software level to form a cluster
- Links commodity hardware with intelligent software to provide application failover and control
- Enables other nodes to take predefined actions when a monitored application fails, for instance to take over and bring up applications elsewhere in the cluster

Figure 2-8 Cluster Server in the guest



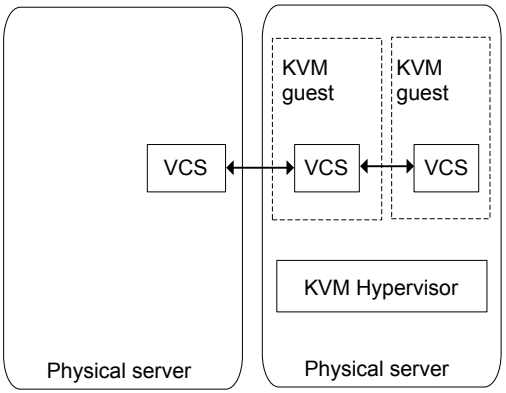
For more information on Cluster Server features, see the *Cluster Server Administrator's Guide*.

Cluster Server in a cluster across virtual machine guests and physical machines

Use Cluster Server (VCS) in both the guest and host to enable an integrated solution for resource management across virtual machines and physical hosts. You can create a physical to virtual cluster combining VCS in a KVM guest together with VCS running on another physical host, enabling VCS to:

- Monitor applications running within the guest
- Failover applications to another physical host
- Failover an application running on a physical host to a VM virtualized guest machine

Figure 2-9 Cluster Server in a cluster across guests and physical machines



For more information on Storage Foundation features, see the *Cluster Server Administrator's Guide*.

Installing Veritas InfoScale Solutions in the kernel-based virtual machine environment

To set up a guest in a kernel-based virtual machine (KVM) environment with Veritas InfoScale Solutions after installing KVM:

Table 2-5 Tasks for installing Veritas InfoScale Solutions in the KVM guest

Task	Information
Set up the KVM host as needed. Create the KVM guests as needed.	See "Creating and launching a kernel-based virtual machine (KVM) host" on page 30. See "Setting up a KVM guest" on page 31.
Install the Veritas InfoScale product on the required KVM guest virtual machines.	For Veritas InfoScale Solutions installation information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.
Configure the SFHA Solutions product on the required KVM guest virtual machines.	For Veritas InfoScale Solutions configuration information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.
Configure resources as required for the KVM guest virtual machines.	See "About kernel-based virtual machine resources" on page 49.

The tasks above apply to the following configurations:

- Dynamic Multi-Pathing in the guest
  See "Dynamic Multi-Pathing in the KVM guest virtualized machine" on page 37.
- Storage Foundation in the guest
  See " Storage Foundation in the virtualized guest machine" on page 38.
- Storage Foundation High Availability in the guest
- Storage Foundation Cluster File System High Availability in the guest
  See " Dynamic Multi-Pathing in the KVM host and Storage Foundation HA in the KVM guest virtual machine" on page 41.

To set up a host in KVM environment with Veritas InfoScale Solutions after installing KVM:

Table 2-6 Tasks for installing Veritas InfoScale Solutions in the KVM host

Task	Information
Configure the KVM host.	See "Creating and launching a kernel-based virtual machine (KVM) host" on page 30.
Install the Veritas InfoScale Solutions product on the KVM host.	For Veritas InfoScale Solutions installation information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.
Configure the Veritas InfoScale Solutions product on the required KVM hosts.	For Veritas InfoScale Solutions configuration information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.
Create the KVM guests as needed.	See "Setting up a KVM guest" on page 31.
Configure resources as required for KVM guest virtual machines.	See "About kernel-based virtual machine resources" on page 49.

The tasks above apply to the following configurations:

- Dynamic Multi-pathing in the host
  See " Dynamic Multi-Pathing in the KVM host" on page 37.
- Storage Foundation Cluster File System High Availability in the host
  See "Storage Foundation Cluster File System High Availability in the KVM host" on page 39.

Installing and configuring Cluster Server in a kernel-based virtual machine (KVM) environment

To set up Cluster Server (VCS) in a KVM environment:

Table 2-7 Tasks for installing VCS in a KVM environment

Task	Information
Set up the KVM host as needed. Create the KVM guests as needed.	See "Creating and launching a kernel-based virtual machine (KVM) host" on page 30. See "Creating and launching a kernel-based virtual machine (KVM) host" on page 30.
Install InfoScale Availability. **Note:** VCS is bundled with the InfoScale Availability product.	For the: *Veritas InfoScale Installation Guide* See "Veritas InfoScale documentation" on page 211.
Configure VCS. No additional VCS configuration is required to make it work inside the guest, provided the host as well as the network are configured.	For the: *Cluster Server Configuration and Upgrade Guide* See "Veritas InfoScale documentation" on page 211.
Configure network as required for KVM guest virtual machines.	See "Network configuration for VM-VM cluster" on page 59.

The steps above apply for the following configurations:

- VCS in the KVM host
  See " Cluster Server in the KVM host" on page 42.
- VCS in the KVM guest
  See "Cluster Server in the guest" on page 43.
- VCS in the KVM host and ApplicationHA in the KVM guest virtual machine
- VCS in a cluster across guests and physical machines
  See "Cluster Server in a cluster across virtual machine guests and physical machines" on page 44.

How Cluster Server (VCS) manages Virtual Machine (VM) guests

High-level overview of how VCS manages VM guests.

- Physical machines form a cluster with VCS installed on them.

For information about installing VCS, see the *Veritas InfoScale Installation Guide*.

- CPU and memory resources are made available to create VM guests on all nodes in the cluster.
- VCS is installed on all the hosts to manage the VM guest.
- The operating system is installed on the VM guest.

Note: The VM guest can be created on an image file or on a shared raw disk, provided the disk names are persistent across all the physical hosts.

- The VM guest is configured as a KVMGuest resource in VCS.

For detailed instructions on creating and configuring a VM guest, see the installation section in the Red Hat Enterprise Linux (RHEL) or SUSE Linux Enterprise Server (SLES) virtualization documentation.

To configure a VM guest for a physical machine to physical machine (PM-PM) configuration, the following conditions apply:

- You must configure a VM guest on one node with operating system installed on a shared storage accessible to all the VCS cluster nodes.
- Ensure that the image file resides on the shared storage so that the virtual machines can fail over across cluster nodes.
- You can configure the first VM guest using the standard installation procedure. See "Installing Veritas InfoScale Solutions in the kernel-based virtual machine environment" on page 45.

Bundled agents are included with VCS for managing many applications. The KVMGuest agent is included and can be used to manage and provide high availability for KVM guests. For information on KVMGuest agent attributes, resource dependency and agent function, refer to the *Cluster Server Bundled Agents Reference Guide*.

Chapter 3

Configuring KVM resources

This chapter includes the following topics:

About kernel-based virtual machine resources

After installing kernel-based virtual machine (KVM) and Veritas InfoScale Solutions products and creating the virtual machines, you can configure your KVM resources to optimize your environment. Configuration processes vary depending on the Veritas InfoScale Solutions you want to configure:

- If you are using Dynamic Multi-Pathing (DMP), Storage Foundation (SF), SFHA, or Storage Foundation Cluster File System High Availability (SFCFSHA) in your guests or hosts, you can optimize your storage for visibility and convenient management.
  See “Configuring storage” on page 50.
- If you are using Cluster Server (VCS), SFHA, or SFCFSHA in your guests or hosts, you can optimize your network to make your KVM resources highly available.
  See “Configuring networking” on page 56.

Configuring storage

Veritas InfoScale Solutions enable you to map and manage your storage more efficiently whether you have a guest or host solution.

Consistent storage mapping in the KVM environment

Managing storage in the KVM environment requires consistent mapping. Storage which is presented to the guest either using the para-virtualized VirtIO drivers, or the fully virtualized IDE emulation, needs to be mapped from the host to the guest. Due to the volatile nature of the device naming used in Linux, care must be taken when mapping storage from the host to the guest. In Linux, the device names are based on enumeration order which can change when systems are rebooted.

Consistent mapping can be achieved by using:

- DMP meta-device
- Mapping devices using device ID
- Mapping devices using paths
- Mapping devices using volumes
- Linux `udev` device sym-links.

Avoid using disk labels when mapping storage to a guest. Disk labels can be modified by a guest and are not guaranteed.

In clustered environments, Active-Passive DMP devices cannot be mapped directly to a guest.

Mapping devices to the guest

Non-persistent mappings can be made using `virsh attach-device`. The non-persistent mappings can be made persistent by redefining the KVM guests using `virsh dumpxml` *`domain`* followed by `virsh define` *`domain`*. Persistent mappings can be created on the host using either `virt-manager` or by modifying the guests XML configuration using `virsh edit` *`domain`*.

The device links created in the directory `/dev/disk/by-path` should be consistent or if possible identical across all the physical hosts. Using different device links can cause issues with virtual machine live migration or VCS KVMGuest Agent failover operations.

See “Mapping devices using the virtio-scsi interface” on page 53.

Mapping DMP meta-devices

Consistent mapping can be achieved from the host to the guest by using the Persistent Naming feature of DMP.

Running DMP in the host has other practical benefits:

- Multi-path device can be exported as a single device. This makes managing mapping easier, and helps alleviate the 32 device limit, imposed by the VirtIO driver.
- Path failover can be managed efficiently in the host, taking full advantage of the Event Source daemon to proactively monitor paths.
- When Veritas InfoScale Solutions products are installed in the guest, the 'Persistent Naming' feature provides consistent naming of supported devices from the guest through the host to the array. The User Defined Names feature, or UDN, allows DMP virtual devices to have custom assigned names.

To map a DMP meta-device to a guest

1. Map the device to the guest. In this example the dmp device *xiv0_8614* is mapped to *guest_1*.

```
# virsh attach-disk guest_1 /dev/vx/dmp/xiv0_8614 vdb
```

2. The mapping can be made persistent by redefining the guest.

```
# virsh dumpxml guest_1 > /tmp/guest_1.xml
# virsh define /tmp/guest_1.xml
```

Consistent naming across KVM Hosts

While enclosure based naming (EBN) provides persistent naming for a single node, it does not guarantee consistent naming across nodes in a cluster. The User Defined Names (UDN) feature of DMP allows DMP devices to be given both persistent and consistent names across multiple hosts. When using User Defined Names, a template file is created on a host, which maps the serial number of the enclosure and device to unique device name. User Defined Names can be manually selected, which can help make mappings easier to manage.

To create consistent naming across hosts

1 Create the User Defined Names template file.

```
# /etc/vx/bin/vxgetdmpnames enclosure=3pardata0 > /tmp/user_defined_names
# cat /tmp/user_defined_names
enclosure vendor=3PARdat product=VV serial=1628 name=3pardata0
 dmpnode serial=2AC00008065C name=3pardata0_1
 dmpnode serial=2AC00002065C name=3pardata0_2
 dmpnode serial=2AC00003065C name=3pardata0_3
 dmpnode serial=2AC00004065C name=3pardata0_4
```

2 If necessary, rename the devices. In this example, the DMP devices are named using the name of the guest they are to be mapped to.

```
# cat /dmp/user_defined_names
enclosure vendor=3PARdat product=VV serial=1628 name=3pardata0
         dmpnode serial=2AC00008065C name=guest1_1
         dmpnode serial=2AC00002065C name=guest1_2
         dmpnode serial=2AC00003065C name=guest2_1
         dmpnode serial=2AC00004065C name=guest2_2
```

3 Apply the User Defined Names file to this node, and all other hosts.

```
# vxddladm assign names file=/tmp/user_defined_names
```

4 Verify the user defined names have been applied.

```
# vxdmpadm getdmpnode enclosure=3pardata0
NAME         STATE     ENCLR-TYPE  PATHS  ENBL  DSBL  ENCLR-NAME
==========================================================================
guest_1_1    ENABLED   3PARDATA    2      2     0     3pardata0
guest_1_2    ENABLED   3PARDATA    2      2     0     3pardata0
guest_2_1    ENABLED   3PARDATA    2      2     0     3pardata0
guest_2_2    ENABLED   3PARDATA    2      2     0     3pardata0
```

Mapping devices using paths

Mapping can be achieved using device ID: /dev/disk/by-path/

These links use the persistent properties of a path. For fibre channel devices, the sym-link name is composed of the bus identifier, the Worldwide Name (WWN) of the target, followed by the LUN identifier. A device will have an entry for each path to the device. In environments where multi-pathing is to be performed in the guest, make a mapping for each path for the device.

In the following example both paths to device *sdd* are mapped to *guest_3*.

To map a path to a guest

1 Identify the devices to map to the guest. Obtain the device IDs.

```
# udevadm info -q symlink --name sdd | cut -d\  -f 3
disk/by-id/scsi-200173800013420cd
```

In multi-path environments the device ID can be used to find all paths to the device.

```
# udevadm info --export-db |grep disk/by-id/scsi-200173800013420cd\  \
| cut -d\  -f 4
/dev/disk/by-path/pci-0000:0b:00.0-fc-0x5001738001340160:0x000000
/dev/disk/by-path/pci-0000:0c:00.0-fc-0x5001738001340161:0x000000
```

2 Map the device to the guest using the path using the device path.

```
# virsh attach-disk guest_3  \
/dev/disk/by-path/pci-0000:0b:00.0-fc-0x5001738001340160:0x000000 vdb
Disk attached successfully
# virsh attach-disk guest_3  \
/dev/disk/by-path/pci-0000:0c:00.0-fc-0x5001738001340161:0x000000 vdc
Disk attached successfully
```

3 Make the mapping persistent by re-defining the guest.

```
# virsh dumpxml guest_3 > /tmp/guest_3.xml
# virsh define /tmp/guest_3.xml
```

Mapping devices using volumes

Mapping can be achieved by using Veritas Volume Manager volumes (VxVM volumes).

For more about mapping a VxVM volume to a guest:

See “Simplified management” on page 118.

Mapping devices using the virtio-scsi interface

In Red Hat Enterprise Linux (RHEL) 6 Update 4 onwards, devices can be mapped to the guest through the virtio-scsi interface, replacing the virtio-blk device and providing the following improvements:

- The ability to connect to multiple storage devices

- A standard command set
- Standard device naming to simplify migrations
- Device pass-through

Note: Mapping using paths is also supported with the virtio-scsi interface.

To enable SCSI passthrough and use the exported disks as bare-metal SCSI devices inside the guest, the `<disk>` element's `device` attribute must be set to "lun" instead of "disk". The following disk XML file provides an example of the `device` attribute's value for virtio-scsi:

```
<disk type='block' device='lun' sgio='unfiltered'>
<driver name='qemu' type='raw' cache='none'/>
<source dev='/dev/disk/by-path/pci-0000:07:00.1-fc-0x5001438011393dee-lun-1'/
<target dev='sdd' bus='scsi'/>
<address type='drive' controller='4' bus='0' target='0' unit='0'/>
</disk>
```

To map one or more devices using virtio-scsi

1. Create one XML file for each SCSI controller, and enter the following content into the XML files:

   ```
   <controller type='scsi' model='virtio-scsi' index='1'/>
   ```

   The XML file in this example is named `ctlr.xml`.

2. Attach the SCSI controllers to the guest:

   ```
   # virsh attach-device guest1 ctlr.xml --config
   ```

3 Create XML files for the disks, and enter the following content into the XML files:

```
<disk type='block' device='lun' sgio='unfiltered'>
<driver name='qemu' type='raw' cache='none'/>
<source dev='/dev/disk/by-path/pci-0000:07:00.1-fc-0x5001438011393dee-lun
<target dev='sdd' bus='scsi'/>
<address type='drive' controller='1' bus='0' target='0' unit='0'/>
</disk>
```

The XML file in this example is named `disk.xml`.

4 Attach the disk to the existing guest:

```
# virsh attach-device guest1 disk.xml --config
```

Resizing devices

Red Hat Linux Enterprise (RHEL) 6.3, 6.4, 6.5, and 7 and SUSE Linux Enterprise Server (SLES) 11 SP2 and SP3 do not support online disk re-sizing of VirtIO devices. To re-size a VirtIO device, the guest must be fully shut down and re-started.

You can use the following methods to resize the devices.

To grow devices

1 Grow the storage.

- If the storage device is a VxVM Volume, re-size the volume.
- If the storage device is a LUN from a storage array, re-size the device on the array.

2 Update the size of the disk device in the host.

- Stop all virtual machines using the storage device.
- If the device is a LUN from a storage array, update the size of the device:

  ```
  # blockdev --rereadpt device
  ```

- Restart the virtual machines.

3 Update the size of the storage device in the guest .

- If VxVM is managing the storage in the guest, use the `vxdisk resize` command.
- If VxVM is not managing the storage in the guest, see the appropriate documentation.

To shrink devices

1. Update the size of the disk device in the guest.
   - If VxVM is managing the device in the guest, if necessary, first use the vxresize utility to shrink any file systems and volumes which are using the device. Use the vxresize utility to update the size of the public region of the device:

     ```
     # vxdisk resize access_name length=new_size
     ```

   - If VxVM is not managing the storage in the guest, see the appropriate documentation.
2. Shrink the storage in the guest.
   - If the device is a VxVM volume, shrink the volume with the vxassist utility.
   - If the device is a LUN from a storage array, shrink the device on storage array.
3. Update the size of the disk device in the host.
   - Stop the guests which are using the devices.
   - If the device is a LUN from a storage array, use the following command:

     ```
     # blockdev --rereadpt device
     ```

4. Start the guests.

Configuring networking

You must configure a network for the host and KVM guest to enable Veritas InfoScale Solutionss to provide:

- Application failover
- Virtual machine availability

Bridge network configuration

The bridge network configuration can be performed in two parts:

- Configuring host network
- Configuring guest network

Host network configuration

The libvirtd service creates a default bridge virbr0 which is a NAT'ed private network. It allocates private IPs from the network 192.168.122.0, to the guests using virbr0 for networking. If the guests are required to communicate on the public network of the host machines, then a bridge must be configured. This bridge can be created using the following steps:

1. Create a new interface file with the name `ifcfg-br0` in `/etc/sysconfig/network-scripts/` location where all the other interface configuration files are present. Its contents are as follows:

   ```
   DEVICE=br0
   Type=Bridge
   BOOTPROTO=dhcp
   ONBOOT=yes
   ```

2. Add the physical interface to the bridge using the following command.

   ```
   # brctl addif eth0 br0
   ```

   This adds the physical interface that the guests shares with the br0 bridge created in the previous step.

3. Verify that your eth0 was added to the br0 bridge using the `brctl show` command.

   ```
   # brctl show
   ```

   The output must look similar to the following:

   ```
   bridge name     bridge id               STP enabled     interfaces
   virbr0          8000.000000000000       yes
   br0             8000.0019b97ec863       yes             eth0
   ```

4. The eth0 network configuration must be changed. The ifcfg-eth0 script is already present.

5. Edit the file and add a line **BRIDGE=br0**, so that the contents of the configuration file look like the following example:

   ```
   DEVICE=eth0
   BRIDGE=br0
   BOOTPROTO=none
   HWADDR=00:19:b9:7e:c8:63
   ONBOOT=yes
   TYPE=Ethernet
   ```

```
USERCTL=no
IPV6INIT=no
PEERDNS=yes
NM_CONTROLLED=no
```

6. Restart the network services to bring all the network configuration changes into effect.

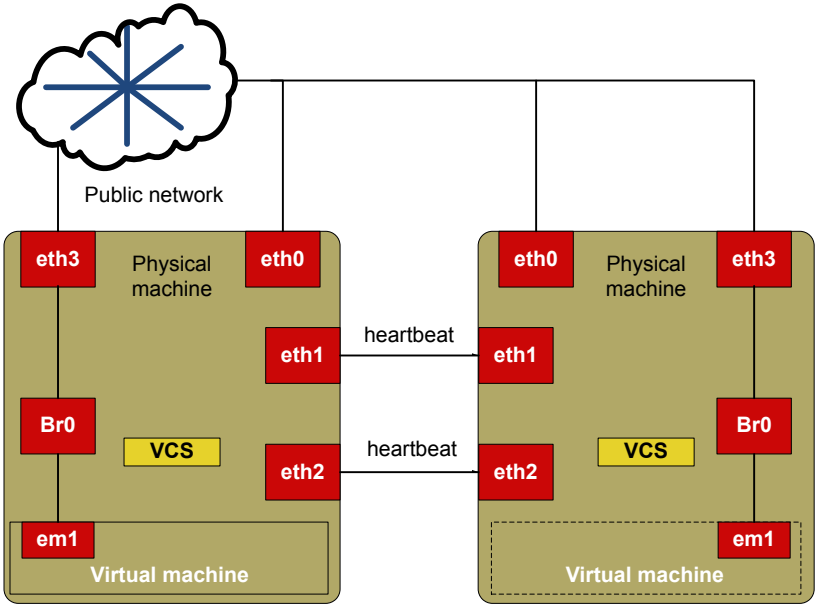
Configuring guest network

Refer to the virtualiztion-related Linux documentation for instructions on configuring guest network.

Network configuration for VCS cluster across physical machines (PM-PM)

The network configuration and storage of the hosts is similar to the VCS cluster configurations. For configuration-related information, refer to the *Veritas InfoScale Installation Guide*. However, you must set up a private link and a shared storage between the physical hosts on which the VM guests are configured.

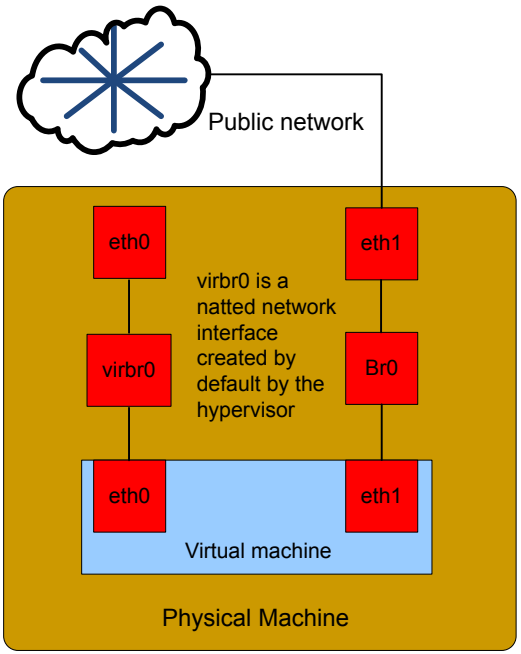
Figure 3-1



Standard bridge configuration

The standard bridge configuration is a generic network configuration for bridge networking.

Figure 3-2 Standard bridge configuration

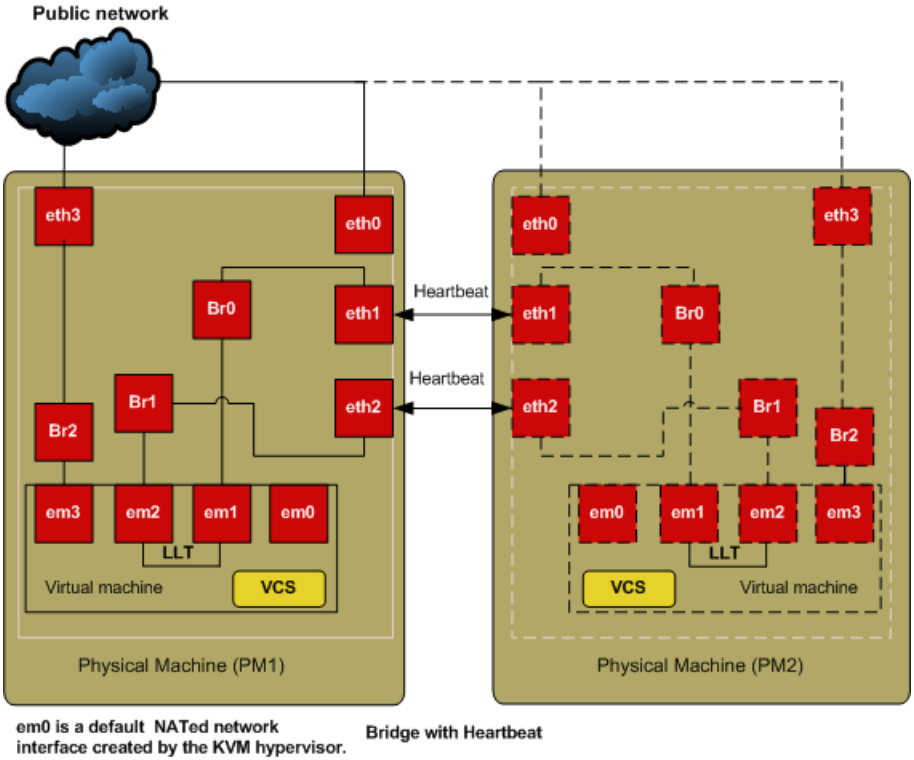


Standard bridge

Network configuration for VM-VM cluster

To configure the VCS cluster between the virtual machines, you must configure the network and storage for the cluster. The setup details for network and storage configurations are explained in the subsequent sections. Figure 3-3 shows a cluster setup between two VM guests running on two different hosts.

Figure 3-3 Network configuration for VM- VM cluster



See “Bridge network configuration” on page 56.

Section 3

Implementing a RedHat Enterprise Virtualization environment

Chapter 4

Getting started with Red Hat Enterprise Virtualization (RHEV)

This chapter includes the following topics:

Creating and launching a RHEV host

Red Hat Enterprise Virtualization (RHEV), an enterprise virtualization product, based on the KVM hypervisor. It provides a centralized virtualization manager with a web interface named RHEV-M, to manage virtual machines on the RHEL-H hosts. RHEV uses Virtual Desktop Server Manager (VDSM) agent in the hosts to manage virtual machine services.

The RHEV-M web interface provides a very simple, easy-to-use and intuitive GUI interface for all virtual machine operations. The features provided by these tools include taking snapshots of virtual machines, creating virtual networks and live migration of virtual machines to another RHEV host.

Once you have configured the required hardware setup:

- Install RHEV on the target systems.
  See "Linux virtualization documentation" on page 212.
- Create and launch the required RHEV virtual machines.
  See "Setting up a virtual machine in the Red Hat Enterprise Virtualization (RHEV) environment" on page 63.
- Proceed to install the required Veritas InfoScale product on the guest or host:
  See "Installing Veritas InfoScale Solutions in the RHEV environment" on page 75.
- Configure VCS for virtual machines
- Configure Storage Foundation as backend storage for virtual machines

Setting up a virtual machine in the Red Hat Enterprise Virtualization (RHEV) environment

Following is a high-level overview of the steps required for setting up virtual machines in Red Hat Enterprise Virtualization (RHEV) environment. For detailed instructions, see the Red Hat Enterprise Virtualization documentation.

To set up virtual machines in the RHEV environment.

1. Before creating virtual machines, ensure that CPU and memory resources are available to create virtual machines on all nodes in the cluster.
2. Make sure that the Virtual Desktop Server Manager (VDSM) service is running on the hosts where virtual machines are to be created. Before you create a virtual machine on a host, make sure that the state of the host in RHEV-M is up.

   ```
   # service vdsmd status
   ```

   Run the command on all the hosts to ensure that the VDSM service is running.
3. Create virtual machines.

   See "Linux virtualization documentation" on page 212.
4. Configure the network for virtual machines.

   See "Network configuration for VM-VM cluster" on page 59.
5. Install the operating system in the virtual machines.
6. Repeat the above steps for all RHEV guests that you want to be a part of the cluster.

Veritas InfoScale Solutions configuration options for the RHEV environment

Veritas InfoScale Solutions products support the configurations listed in Table 4-1. The configurations profiled in the table below are the minimum required to achieve the storage and availability objectives listed. You can mix and match the use of Veritas InfoScale Solutions products as needed to achieve the desired level of storage visibility, management, replication support using VVR, availability, and cluster failover for your RHEV-based virtual machines (RHEV) hosts and guest virtual machines.

Table 4-1 Veritas InfoScale Solutions supported configuration options in the RHEV environment

Objective	Recommended Veritas InfoScale Solutions product configuration
Storage visibility for RHEV guest virtual machines	Dynamic Multi-Pathing (DMP) in the RHEV guest virtual machines
Storage visibility for RHEV hosts	DMP in the RHEV hosts
Storage management features and replication support using VVR for RHEV guest virtual machines	Storage Foundation (SF) in the RHEV guest virtual machines See " Storage Foundation in the RHEV guest virtual machine" on page 66.
Advanced storage management features and replication support using VVR for RHEV hosts	Storage Foundation Cluster File System (SFCFSHA) in the RHEV hosts
End-to-end storage visibility in RHEV hosts and guest virtual machines	DMP in the RHEV host and guest virtual machines
Storage management features and replication support using VVR in the RHEV guest virtual machines and storage visibility in in the RHEV host	DMP in the RHEV host and SF in the RHEV guest virtual machines See " Dynamic Multi-Pathing in the RHEV host and Storage Foundation HA in the RHEV guest virtual machine" on page 69.
Virtual machine monitoring, migration, and failover for RHEV hosts	Cluster Server (VCS) in the RHEV hosts
Application failover for RHEV guest virtual machines	VCS in the RHEV guest virtual machines

Table 4-1 Veritas InfoScale Solutions supported configuration options in the RHEV environment *(continued)*

Objective	Recommended Veritas InfoScale Solutions product configuration
Application failover across RHEV guest virtual machines and physical hosts	VCS in RHEV guest virtual machines and RHEV physical host machines

See "About setting up RHEV with Veritas InfoScale Solutions" on page 72.

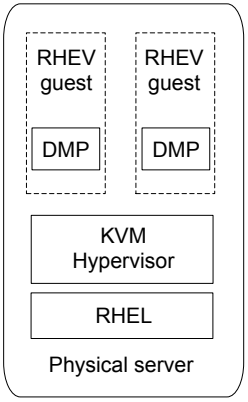
See "Virtualization use cases addressed by Veritas InfoScale products" on page 22.

Dynamic Multi-Pathing in a RHEV guest virtual machine

Use Dynamic Multi-Pathing (DMP) to provide storage visibility in RHEV guest virtualized machines. DMP in the RHEV guest virtualized machine provides:

- Multi-pathing functionality for the operating system devices configured in the guest
- DMP metadevices (also known as DMP nodes) to represent all the device paths to the same physical LUN
- Support for enclosure-based naming
- Support for standard array types

Figure 4-1 Dynamic Multi-Pathing in the guest



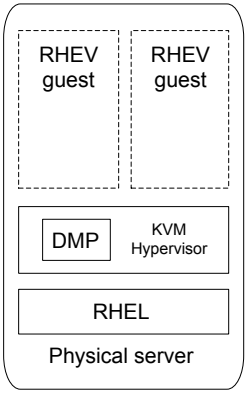
For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

Dynamic Multi-Pathing in the RHEV host

Use Dynamic Multi-Pathing (DMP) to provide storage visibility in the RHEV hosts. Using DMP in the RHEV host enables:

- Centralized multi-pathing functionality
- Enables active/passive array high performance failover
- Centralized storage path management
- Fast proactive failover
- Event notification

Figure 4-2 Dynamic Multi-Pathing in the RHEV host



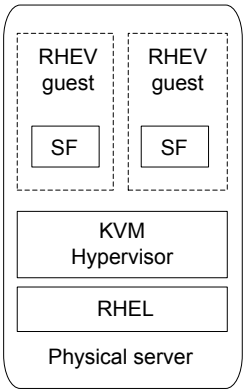
For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

Storage Foundation in the RHEV guest virtual machine

Use Storage Foundation (SF) in the guest to provide storage management functionality for KVM guest virtual machine resources. Storage Foundation enables you to manage KVM guest storage resources more easily by providing:

- Enhanced database performance
- Point-in-time copy features for data back-up, recovery, and processing
- Options for setting policies to optimize storage
- Methods for migrating data easily and reliably
- Replication support

Figure 4-3 Storage Foundation in the virtualized guest machine



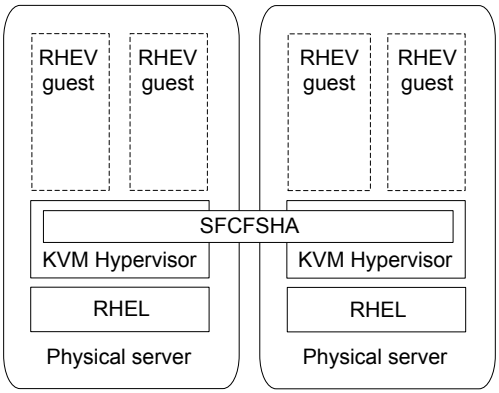
For more information on Storage Foundation features, see the *Storage Foundation Administrator's Guide*.

Storage Foundation Cluster File System High Availability in the RHEV host

Use Storage Foundation Cluster File System High Availability (SFCFSHA) to provide advanced storage management functionality for the RHEV host. SFCFSHA enables you to manage your RHEV host storage resources more easily by providing:

- Enhanced database performance
- Point-in-time copy features for data back-up, recovery, and processing
- Options for setting policies to optimize storage
- Methods for migrating data easily and reliably
- Replication support
- High availability for virtual machines
- High availability and disaster recovery for virtual machines
- Simplified management of virtual machines

Figure 4-4 Storage Foundation Cluster File System High Availability in the RHEV host



For more information on Storage Foundation features, see the *Storage Foundation™ Cluster File System High Availability Administrator's Guide*.

Dynamic Multi-Pathing in the RHEV host and guest virtual machine

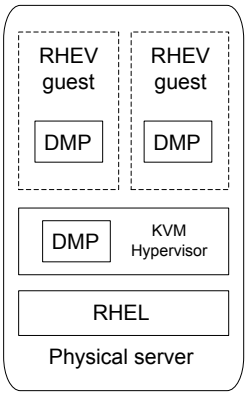
Use Dynamic Multi-Pathing (DMP) to provide end-to-end storage visibility across both the RHEV host and guest virtual machine. Using DMP in the RHEV guest virtualized machine provides:

- Multi-pathing functionality for the operating system devices configured in the guest
- DMP metadevices (also known as DMP nodes) to represent all the device paths to the same physical LUN
- Support for enclosure-based naming
- Support for standard array types

Using DMP in the RHEV host enables:

- Centralized multi-pathing functionality
- Enables active/passive array high performance failover
- Centralized storage path management
- Fast proactive failover
- Event notification

Figure 4-5 Dynamic Multi-Pathing in the KVM virtualized guest and the KVM host



For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

Dynamic Multi-Pathing in the RHEV host and Storage Foundation HA in the RHEV guest virtual machine

Use Storage Foundation and High Availability (SFHA) in the guest in combination with Dynamic Multi-Pathing (DMP) in the RHEV host to combine storage management functionality for RHEV guest virtual machine resources and storage visibility in the RHEV host.

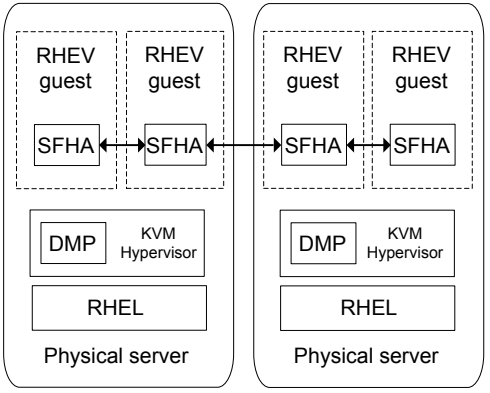
Using SFHA in the RHEV guest provides:

- Enhanced database performance
- Point-in-time copy features for data back-up, recovery, and processing
- Options for setting policies to optimize storage
- Methods for migrating data easily and reliably
- Replication support
- High availability for applications running inside virtual machines

Using DMP in the RHEV host provides:

- Centralized multi-pathing functionality
- Fast proactive failover.
- Event notification

Figure 4-6 Storage Foundation HA in the RHEV guest virtual machine and DMP in the RHEV host



For more information on SFHA features, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

For more information on DMP features, see the *Dynamic Multi-Pathing Administrator's Guide*.

Cluster Server for the RHEV environment

Cluster Server (VCS) provides virtual machine monitoring and failover to another host in the Red Hat Enterprise Virtualization (RHEV) environment. VCS enables the following for RHEV hosts:

- Connects multiple, independent systems into a management framework for increased availability.
- Enables nodes to cooperate at the software level to form a cluster.
- Links commodity hardware with intelligent software to provide application failover and control.
- Enables other nodes to take predefined actions when a monitored application fails, for instance to take over and bring up applications elsewhere in the cluster.

VCS supports the following configurations:

Table 4-2 VCS supported configuration options in the RHEV environment

Objective	Recommended VCS configuration
Virtual machine monitoring and failover for hosts	VCS in the hosts

Table 4-2 VCS supported configuration options in the RHEV environment *(continued)*

Objective	Recommended VCS configuration
Disaster recovery in virtualized environment	VCS on the Red Hat Enterprise Linux (RHEL) hypervisor
Application failover for guest virtual machines	VCS in the guest virtual machines
Application failover across guest virtual machines and physical hosts	VCS in guest virtual machines and physical host machines

Note: Virtual machine disaster recovery is supported in the RHEV environment only

Figure 4-7 Cluster Server in the RHEV host

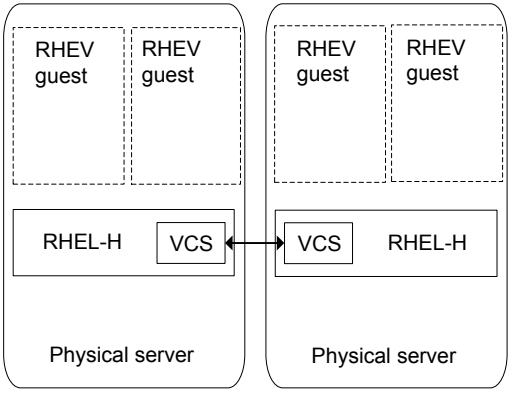
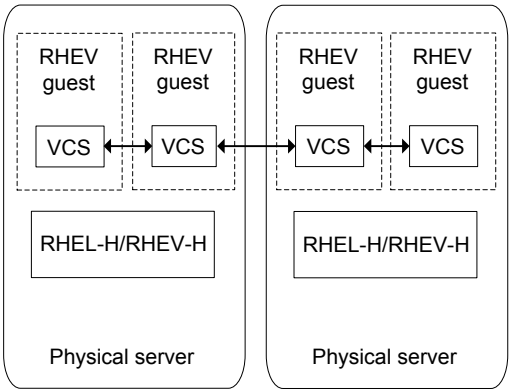


Figure 4-8 Cluster Server in the RHEV guest



For more information on VCS features, see the *Cluster Server Administrator's Guide*.

About setting up RHEV with Veritas InfoScale Solutions

Before setting up your virtual environment, verify that your planned configuration will meet the system requirements, licensing and other considerations for installation with Veritas InfoScale Solutions products.

- Licensing: customers running Storage Foundation (SF) or Storage Foundation Cluster File System High Availability (SFCFSHA) in a RHEV-based virtual machine (RHEV) environment are entitled to use an unlimited number of guests on each licensed server or CPU.
- Red Hat system requirements: see Table 4-3
- Veritas product requirements: see Table 4-4
- *Release Notes*: each Veritas product contains last minute news and important details for each product, including updates to system requirements and supported software. Review the *Release Notes* for the latest information before you start installing the product.

  The product documentation is available on the Web at the following location: https://sort.veritas.com/documents

Table 4-3 Red Hat and SUSE system requirements

	Red Hat Enterprise Linux (RHEL)
Supported architecture	■ Intel 64 ■ AMD 64
Minimum system requirement	■ 6 GB free disk space ■ 2 GB of RAM
Recommended system requirement	■ 6 GB plus the required disk space recommended by the guest operating system per guest. For most operating systems more than 6 GB of disk space is recommended ■ One processor core or hyper-thread for each virtualized CPU and one for the host ■ 2 GB of RAM plus additional RAM for virtualized guests
Hardware requirement	Full virtualization-enabled CPU
Veritas InfoScale Solutions	Veritas InfoScale Solutions 7.4.1
Supported OS version in the host	RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 4, Update 5, Update 6 SLES11 SP3, SP4 SLES12 SP2, SP3
Supported OS version in the virtual machine	RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 4, Update 5, Update 6 SLES11 SP3, SP4 SLES12 SP2, SP3
Supported RHEV version	RHEV 3.5, 3.6

Table 4-4 Veritas product requirements

Hardware	https://www.veritas.com/support/en_US/article.000126344

Table 4-4 Veritas product requirements *(continued)*

Software	■ Dynamic Multi-Pathing 7.4.1 Used for storage visibility on RHEV hosts and guest virtual machines ■ Storage Foundation 7.4.1 Used for storage management on RHEV hosts and guest virtual machines ■ Storage Foundation HA 7.4.1 Used for storage management and clustering on RHEV hosts and guest virtual machines ■ Storage Foundation Cluster File System High Availability 7.4.1 Used for storage management and clustering multiple RHEV hosts to enable live migration of guest virtual machines ■ Cluster Server 7.4.1 Used for virtual machine monitoring, migration, and failover ■ Veritas Operations Manger 5.0 Used for application visibility and virtual host management
Storage	■ Shared storage for holding the guest image. (VM failover) ■ Shared storage for holding the application data. (Application failover) ■ Local storage with Flexible Shared Storage (FSS) option for VM and application failover
Networking	■ Configure the guest for communication over the public network ■ Setup virtual interfaces for private communication.
Documentation: see the product release notes to for the most current system requirements, limitations, and known issues:	■ *Veritas InfoScale Release Notes* ■ Veritas Services and Operations Readiness Tools (SORT): https://sort.veritas.com/documents

Table 4-5 VCS system requirements for RHEV-supported Red Hat Enterprise Linux configurations

VCS version	7.4.1
Supported OS version in host	RHEL 6 Update 8, Update 9, Update 10 RHEL 7 Update 4, Update 5, Update 6 SLES11 SP3, SP4 SLES12 SP2, SP3

Table 4-5 VCS system requirements for RHEV-supported Red Hat Enterprise Linux configurations *(continued)*

Supported OS in VM guest	RHEL 6 Update 8, Update 9, Update 10
	RHEL 7 Update 4, Update 5, Update 6
	SLES11 SP3, SP4
	SLES12 SP2, SP3
Hardware requirement	Full virtualization-enabled CPU

Limitations and unsupported RHEV-based virtual machine features

For more information on limitations and known issues, see the *Veritas InfoScale Release Notes* for Linux.

For RHEV related limitations, see the Virtualization technology provider (RHEL) release notes.

Installing Veritas InfoScale Solutions in the RHEV environment

To set up a guest in a RHEV-based virtual machine (RHEV) environment with Veritas InfoScale Solutions after installing RHEV:

Table 4-6 Tasks for installing Veritas InfoScale Solutions in the RHEV guest

Task	Information
Set up the RHEV host as needed. Create the RHEV guests as needed.	See "Creating and launching a RHEV host" on page 62. See "Setting up a virtual machine in the Red Hat Enterprise Virtualization (RHEV) environment" on page 63.
Install the Veritas InfoScale Solutions product on the required RHEV guest virtual machines.	For Veritas InfoScale Solutions installation information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.
Configure the Veritas InfoScale Solutions product on the required RHEV guest virtual machines.	For Veritas InfoScale Solutions configuration information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.

Table 4-6 Tasks for installing Veritas InfoScale Solutions in the RHEV guest *(continued)*

Task	Information
Configure resources as required for the RHEV guest virtual machines.	

The tasks above apply to the following configurations:

- Dynamic Multi-Pathing in the guest
  See "Dynamic Multi-Pathing in a RHEV guest virtual machine" on page 65.
- Storage Foundation in the guest
  See " Storage Foundation in the RHEV guest virtual machine" on page 66.
- Storage Foundation High Availability in the guest
- Storage Foundation Cluster File System High Availability in the guest
  See "Storage Foundation Cluster File System High Availability in the RHEV host" on page 67.

To set up a host in RHEV environment with Veritas InfoScale Solutions after installing RHEV:

Table 4-7 Tasks for installing Veritas InfoScale Solutions in the RHEV host

Task	Information
Configure the RHEV host.	See "Creating and launching a RHEV host" on page 62.
Install the Veritas InfoScale Solutions product on the RHEV host.	For Veritas InfoScale Solutions installation information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.
Configure the Veritas InfoScale Solutionss product on the required RHEV hosts.	For Veritas InfoScale Solutions configuration information, see the product installation guides. See "Veritas InfoScale documentation" on page 211.
Create the RHEV guests as needed.	See "Setting up a virtual machine in the Red Hat Enterprise Virtualization (RHEV) environment" on page 63.
Configure resources as required for RHEV guest virtual machines.	

The tasks above apply to the following configurations:

- Dynamic Multi-pathing in the host
  See " Dynamic Multi-Pathing in the RHEV host" on page 66.
- Storage Foundation Cluster File System High Availability in the host
  See "Storage Foundation Cluster File System High Availability in the RHEV host" on page 67.

Chapter 5

Configuring VCS to manage virtual machines

This chapter includes the following topics:

Installing and configuring Cluster Server for virtual machine and application availability

To set up Cluster Server (VCS) in Red Hat Enterprise Virtualization (RHEV) environment:

- Install VCS.
- Configure VCS.

How Cluster Server (VCS) manages virtual machines

Following is a high-level overview of how VCS manages virtual machines in the Red Hat Enterprise Virtualization (RHEV) environment:

- Physical machines form a cluster with VCS installed on them.
  See the *Veritas InfoScale Installation Guide* for installation information.

- CPU and memory resources are made available to host virtual machines on all nodes in the cluster.
- VCS is installed on all the hosts to manage the virtual machines.
- The operating system is installed on the virtual machine on any one host.
- The virtual machine is configured as a KVMGuest resource in VCS.

About the KVMGuest agent

The KVMGuest agent enables Cluster Server (VCS) to monitor a KVM guest - that is, a virtual machine in the KVM environment or the Red Hat Enterprise Virtualization (RHEV) environment. The agent performs tasks such as bringing virtual machines online and taking them offline. The KVMGuest agent operates in both KVM and RHEV environments. This topic describes its behavior in the RHEV environment.

For details on the KVMGuest agent behavior in open source KVM environment or RHEV environment, see the *Cluster Server Bundled Agents Reference Guide*.

The KVMGuest agent uses `virsh` commands to manage virtual machines in the KVM environment and Representational State Transfer (REST) APIs to manage virtual machines in RHEV environment by using the REST APIs to determine the state of the virtual machine. The agent determines the resource state, on the basis of the virtual machine state. REST design architecture focuses on resources and their representations for some specific service. REST APIs help software developers and administrators integrate the functionality of the RHEV environment with custom scripts or with external applications which access the API by means of HTTP.

Prerequisites for administering virtual machines in a RHEV environment by using REST APIs:

- A networked installation of Red Hat Enterprise Virtualization Manager, which includes the REST API
- A client or programming library that initiates and receives HTTP requests from the REST API

The following table lists various states of a virtual machine in RHEV environment and the corresponding VCS resource state:

Table 5-1

Virtual machine state	VCS resource state	Resource confidence level
wait_for_launch	ONLINE	10
powering_up	ONLINE	60

Table 5-1 *(continued)*

Virtual machine state	VCS resource state	Resource confidence level
up	ONLINE	100
powering_down	ONLINE	40
paused	ONLINE	20
down	OFFLINE	–
saving_state	INTENTIONAL OFFLINE	–
suspended	INTENTIONAL OFFLINE	–
restoring_state	ONLINE	50
migrating	INTENTIONAL OFFLINE	–
reboot_in_progress	INTENTIONAL OFFLINE	–
image_locked	UNKNOWN	–
unknown	UNKNOWN	–

Table 5-2 KVMGuest agent functions

Function	Tasks
Online	KVM environment: Agent uses the `virsh start` command to start the guest virtual machine. When the resource is configured to define the guest configuration, agent uses the `virsh define` command to define the virtual machine while bringing it online. RHEV environment: Agent uses the REST APIs to start the virtual machine. If the DROpts attribute is set to configure the guest network, the agent also sets the payload as a cdrom. This payload contains networking parameters to be set within the guest after a DR failover. The agent waits for a certain time period after initiating the virtual machine start. You can specify this wait period by using the “DelayAfterGuestOnline” attribute. The agent also checks whether the virtual machine is configured for disaster recovery by checking the `DROpts` attribute. If this attribute is set correctly, the agent inserts a virtual CDROM into the virtual machine's configuration. This CDROM contains a file that contains the site-specific network parameters to be applied at this site for the virtual machine. When the virtual machine boots, the `vcs-net-reconfig` service installed inside the guest checks for the CDROM and the disaster recovery information. If the `vcs-net-reconfig` service finds the disaster recovery information, the service applies the networking parameters to the virtual machine.
Offline	The Offline function initiates a graceful shutdown of the virtual machine. KVM environment: Agent uses the `virsh shutdown` command to shutdown the guest virtual machine. If the SyncDir attribute is configured to synchronize the guest virtual machine configuration file, then the configuration file is copied to the location configured as a SyncDir attribute. RHEV environment: Agent uses the REST APIs to shutdown the virtual machine. The agents wait for a certain time period after initiating the shutdown for the virtual machine to shut down completely. You can specify this wait period by using the DelayAfterGuestOffline attribute.

Table 5-2 KVMGuest agent functions *(continued)*

Function	Tasks
Monitor	KVM environment: Agent uses the `virsh domstate` command to determine the status of the guest virtual machine. RHEV environment: Agent uses the REST APIs to get the status of the virtual machine.
Clean	KVM environment: Agent uses the `virsh destroy` command to forcefully shutdown the guest virtual machine. If the SyncDir attribute is configured to synchronize the guest virtual machine configuration file, then the configuration file is copied to the location configured as a SyncDir attribute. RHEV environment: Agent uses REST APIs to stop the virtual machine.
Migrate	KVM environment: The agent uses the `virsh migrate` command to start virtual machine migration. RHEV environment: The agent uses REST APIs to start virtual machine migration. Additionally, it checks whether the virtual machine migration is allowed or not. **Note:** When a virtual machine is configured for disaster recovery, the virtual machine cannot be migrated across sites.

The KVMGuest agent recognizes the following resource states:

Table 5-3

Resource state	Description
ONLINE	Indicates that the guest virtual machine is running.
OFFLINE	Indicates that the guest virtual machine has stopped.
FAULTED	Indicates that the guest virtual machine has failed to start or has unexpectedly stopped.
UNKNOWN	Indicates that a problem exists with the configuration or with the ability to monitor the resource.
INTENTIONAL OFFLINE	Indicates that the virtual machine has either migrated to another physical host or the administrator intentionally suspended it.

The Cluster Server agent for managing virtual machines in a KVM or RHEV environment, is represented by the KVMGuest resource type definition:

```
type KVMGuest (
        static int IntentionalOffline = 1
        static boolean AEPTimeout = 1
        static int MigrateTimeout = 300
        static int MigrateWaitLimit = 2
        static keylist SupportedActions = { "guestmigrated", "vmconfigsync", "DevScan" }
        static keylist SupportedOperations = { "migrate"}
        static keylist RegList = { "GuestName", "DelayAfterGuestOnline", "DelayAfterGuestOffline
        static str ArgList[] = { GuestName, DelayAfterGuestOnline, DelayAfterGuestOffline, SyncD
        str CEInfo{} = { Enabled=0, CESystem=NONE, FaultOnHBLoss=1 }
        str RHEVMInfo{} = { Enabled=0, URL=NONE, User=NONE, Password=NONE, Cluster=NONE, UseManu
        str GuestName
        int DelayAfterGuestOnline = 5
        int DelayAfterGuestOffline = 30
        str SyncDir
        str GuestConfigFilePath
        boolean ResyncVMCfg = 0
        str DROpts{} = { ConfigureNetwork=0, IPAddress=NONE, Netmask=NONE, Gateway=NONE, DNSServ
)
```

The `RHEVMInfo` attribute enables the KVMGuest attribute configuration to support the Red Hat Enterprise Virtualization environment. RHEVMInfo specifies the following information about the RHEV environment:

Attribute value	Description
Enabled	Specifies whether the virtualization environment is a KVM environment or a Red Hat Enterprise Virtualization (RHEV) environment. 0 indicates the KVM environment. 1 indicates the RHEV environment. The default value is 0.
URL	Specifies the RHEV-M URL, that the KVMGuest agent can use for REST API communication. The API can only communicate with the secure port (SSL). For example: `https://rhevm-server.example.com:443`

Attribute value	Description

User	Specifies the RHEV-M user name that the agent must use for REST API communication. For example:
	`admin@internal`
	`rhevadmin@example.com`

Password	Specifies the encrypted password associated with the RHEVM user profile. Use the `vcsencrypt` utility to encrypt the password. For details, see the *Cluster Server Administrator's Guide*.
Cluster	Specifies the name of the RHEV-M cluster of which the VCS host is a member.
UseManualRHEVMFencing	Specifies if the use of manual RHEV-M fencing is enabled in the event that the physical host on which virtual machine is running crashes. 0 indicates that manual RHEV-M fencing is disabled. 1 indicates that manual RHEV-M fencing is enabled. The default value is 0.

The `DROpts` attribute enables the virtual machine for disaster recovery. The attribute contains site-specific network parameters for the virtual machine. The value of this attribute consists of the following keys that define the disaster recovery options for the virtual machine:

Attribute keys	Description
DNSSearchPath	The domain search path used by the virtual machine in this site. The value of this key must contain a list of DNS domain names that are used for the DNS lookup of a hostname in case the domain name of the hostname is not specified. Use spaces to separate the domain names.
DNSServers	The list of DNS servers used by the virtual machine in this site. The value of this key must contain a list of IP addresses of DNS servers that are used for the DNS lookup of a hostname. Use spaces to separate the IP addresses.
Gateway	The default gateway used by the virtual machine in this site.
Device	The Network Interface Card (NIC) that is dedicated to the exclusive IP address of the virtual machine in this site. If this key is not specified, the agent automatically selects the first dedicated NIC for the assignment of the IP address, if specified. Example: eth0.

Attribute keys	Description
IPAddress	The IP address to be assigned to the virtual machine in this site after a cross-site failover.
Netmask	The netmask to be used by the virtual machine in this site after a cross-site failover.
ConfigureNetwork	The `DROpts` attribute value is applied to the virtual machine only if this key is set to 1. Type and dimension: string-association.

Note: For information on other attributes associated with the KVMGuest agent, see the *Cluster Server Bundled Agents Reference Guide*.

Validating the virtualization environment

The KVMGuest agent validates the virtualization environment with the help of a standalone utility `havirtverify`.

The agent invokes this utility in `open` entry point and `attr_changed` entry point. The utility validates the configured virtualization environment for a resource based on its configuration.

For RHEV, the utility:

- Validates the configured URL and user credentials.
- Verifies whether RHEV HA for a configured virtual machine is disabled or not.
- Verifies the `DROpts` attribute

For KVM, the utility checks whether `libvirtd` is running or not.

Once the validation is passed, the agent can start monitoring the resource. If validation fails for a particular resource, its state is reported as UNKNOWN. This validation is also triggered if value of either of the following attributes changes:`RHEVMInfo`, `GuestName`.

You can also run this utility manually for verifying the environment.

To validate the RHEV environment

- Run:

  ```
  # /opt/VRTSvcs/bin/KVMGuest/havirtverify resource_name
  ```

  If validation passes, the following message displays:

  ```
  #/opt/VRTSvcs/bin/KVMGuest/havirtverify resource_name
  Red Hat Enterprise Virtualization Environment validation successfully
  completed for resource resource_name
  ```

  If validation fails, the following message displays:

  ```
  # /opt/VRTSvcs/bin/KVMGuest/havirtverify resource_name
  Virtualization environment validation failed for resource resource_name
  ```

All the log messages of this utility are sent to the engine log file.

Configuring a resource in a RHEV environment

Before you configure a resource in a RHEV environment, you must:

- Ensure that RHEV-HA is disabled for the virtual machine which you want to configure monitoring with Cluster Server (VCS).
- Configure the virtual machine to run on a specific host and the virtual machine image must be available to all the hosts in the VCS cluster.
- Configure the firewall settings to allow REST API communication.

To configure a KVMGuest resource

1. Validate the virtualization environment.

   See "Validating the virtualization environment" on page 85.

2. Specify the name of the virtual machine that VCS must manage, as the value of the GuestName attribute.
3. Configure the DelayAfterGuestOnline and DelayAfterGuestOffline attributes.

   Note: The default value of DelayAfterGuestOnline is 5 and DelayAfterGuestOffline is 30.

4 Validate the RHEV-M URL, valid RHEV-M user (name), and password.

5 To configure the RHEVMInfo attribute, specify the appropriate value of each key. The following table lists each key and its related instruction:

Key	Instruction
Enabled	Set the value to 1.
URL	Specify the RHEV-M URL.
User	Specify a valid user name. For example: admin@internal rhevadmin@example.com
Password	Specify the encrypted password associated with RHEV-M User profile. For details on encrypting passwords, see the *Cluster Server Administrator's Guide*.
Cluster	Specify the RHEV-M cluster name.
UseManualRHEVMFencing	Enable the use of manual RHEV-M fencing in the event that the physical host on which virtual machine is running crashes. For example: # **`UseManualRHEVMFencing=1`**

Configuring multiple KVMGuest resources

If a VCS service group has more than one KVMGuest resource monitoring virtual machines and one of the virtual machines is migrated to another host, then a service group level concurrency violation occurs as the service group state goes into PARTIAL state on multiple nodes.

Veritas recommends configuring only one KVMGuest resource in a Service group. See the sample configurations below for reference.

Configuration 1:

```
group rhev_grp1 (

SystemList = { sys1 = 0, sys2 = 1 }
)

KVMGuest kvmres1 (
```

```
RHEVMInfo = { Enabled = 1,

URL = "https://rhevm-server.example.com:443",

User = "admin@internal"

Password = bncNfnOnkNphChdHe,

Cluster = dc2_cluster1,

UseManualRHEVMFencing=1 }

GuestName = rhevvm1

DelayAfterGuestOnline = 20

DelayAfterGuestOffline = 35

)
```

Configuration 2:

```
group rhev_grp1 (

SystemList = { sys1 = 0, sys2 = 1 }
)

KVMGuest kvmres1 (

RHEVMInfo = { Enabled = 1,

URL = "https://rhevm-server.example.com:443",

User = "admin@internal"

Password = bncNfnOnkNphChdHe,

Cluster = dc2_cluster1,

UseManualRHEVMFencing=0 }

GuestName = rhevvm1

DelayAfterGuestOnline = 20
```

```
DelayAfterGuestOffline = 35

)

group rhev_grp2 (

SystemList = { sys1 = 0, sys2 = 1 }
)

KVMGuest kvmres2 (

RHEVMInfo = { Enabled = 1,

URL = "https://rhevm-server.example.com:443",

User = "admin@internal"

Password = bncNfnOnkNphChdHe,

Cluster = dc2_cluster1,

UseManualRHEVMFencing=0 }

GuestName = rhevvm2

DelayAfterGuestOnline = 20

DelayAfterGuestOffline = 35

)
```

Chapter 6

Configuring Storage Foundation as backend storage for virtual machines

This chapter includes the following topics:

About configuring virtual machines to attach Storage Foundation as backend storage in an RHEV environment

The backend storage for guest virtual machines in an RHEV environment can be derived from Storage Foundation (SF) components. SF as a storage management solution provides enterprise class storage management in comparison to the native logical volume manager and file system solutions. Storage for virtual machines can be configured after you install and configure SF components on RHEL-H hosts. Install the VRTSrhevm package on the RHEV Manager (RHEV-M), to enable the Storage Foundation Administration utility for RHEV. Run the utility on RHEV-M host to manage virtual machines.

After you configure storage for virtual machines, the exported Storage Foundation components are visible as SCSI-3 devices, cluster wide. Note that virtual machines can view only the DMP attributes but it cannot view the volume and file attributes because SF is installed on the host and not on the guest.

Evaluate the type of storage network you want to deploy. You can choose between either a SAN network or commodity storage array by leveraging Flexible shared storage (FSS) capability available in SFCFSHA or SFHA components. Using FSS means that storage may be local to each RHEL-H host. However the local storage is shared over the network for CVM and CFS. FSS potentially lets you deploy a SAN-free environment. It also scales the storage capacity vertically (memory, CPU, and so ont) and horizontally (multiple local storage arrays), each host serving both compute and storage needs.

Use cases for virtual machines using Storage Foundation storage

Table 6-1 Use cases and recommended Veritas InfoScale Solutions

Use case	Recommended Veritas InfoScale Solutions	Storage
Live migration in an RHEV environment	SFCFSHA or SFHA on host	SAN network or FSS enabled with commodity local storage hardware

Table 6-1 Use cases and recommended Veritas InfoScale Solutions *(continued)*

Use case	Recommended Veritas InfoScale Solutions	Storage
Disaster recovery in an RHEV environment	SFCFSHA or SFHA on host, VVR, VFR replication that is configured with VCS Global Cluster Option	SAN network or FSS enabled with commodity local storage hardware
Storage scale out horizontally (connecting multiple local storage that act as a shared storage)	SFCFSHA on host	FSS with commodity local storage

Workflow to configure storage for virtual machines in an RHEV environment

Table 6-2 Tasks to configure storage for virtual machines in an RHEV environment

Workflow task	Details
Prerequisites	Network connectivity, OS version, communication mode between RHEL-H and RHEV-M. See "Prerequisites in an RHEV environment" on page 93.
Installing Storage Foundation administration utility on RHEV-M	See "Installing the SF administration utility for RHEV" on page 93.
Installing Veritas InfoScale Enterprise and configuring SFCFSHA or SFHA on RHEL-H hosts	See "Installing and configuring SFCFSHA or SFHA cluster on RHEL-H nodes" on page 93.
Administer FSS to use commodity storage hardware	See "Installing and configuring SFCFSHA or SFHA cluster on RHEL-H nodes" on page 93.
Configuring virtual machines for Storage Foundation components	See "Configuring Storage Foundation as backend storage for virtual machines" on page 94.

Table 6-2 Tasks to configure storage for virtual machines in an RHEV environment *(continued)*

Workflow task	Details
Usage examples from the RHEV administration utility (vxrhevadm utility)	See "Usage examples from the RHEV administration utility" on page 97.

Prerequisites in an RHEV environment

- Ensure that RHEV-M and RHEL-H hosts are connected over a network.
- Ensure that RHEL-H hosts run RHEL 6 in an RHEV 3.x environment.
- Password-less SSH communication is set up between the RHEV-M and all the RHEL-H hosts. The RHEV-M manager must have root user access to the all the hosts.

Installing the SF administration utility for RHEV

After you complete installation of REHV-M, install the `VRTSrhevm` package available from the Veritas InfoScale products install bundle source nodes where the RHEV-M service is running. This package enables a command line interface from RHEV-M to attach Storage Foundation components to individual virtual machines.

In a highly available cluster or disaster recovery configuration for RHEV-M, ensure that the `VRTSrhevm` package is installed on the nodes where the RHEV-M service is running. This package is required on the RHEV-M node to administer virtual machines.

```
# rpm -ivh VRTSrhevm-6.2.0.000-GA_RHEL6.x86_64.rpm
```

For more information, refer to Linux Red Hat documentation.

Installing and configuring SFCFSHA or SFHA cluster on RHEL-H nodes

Install and configure a standard installation of the SFCFSHA cluster or SFHA on the RHEL-H hosts.

Installing SFCFSHA automatically enables Flexible Storage Sharing (FSS). No additional steps are required. LLT, GAB, and I/O fencing must be configured before administering FSS.

For more information on installing Veritas InfoScale products , refer to the *Veritas InfoScale Installation Guide*

SFCFSHA: For more information on administering FSS and performing other administrative tasks, refer to the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

Configuring Storage Foundation as backend storage for virtual machines

Configure virtual machines from the `VRTSrhevm` utility that is installed on the RHEV-M machine. Access the utility at, `/opt/VRTSrhevm/bin/vxrhevadm`.

Configuration tasks include attach or detach SF components to individual virtual machines, start and stop virtual machines, configure attached devices from a host, and view the assigned storage devices to the virtual machine.

Attaching or detaching Storage Foundation components in guest virtual machines

Aattach or detach DMP device, volume device, or Veritas File System (VxFS) file as the storage backend for the specified virtual machine through the SF administration utility from the RHEV-M machine. These attached devices should be a shared entity across RHEV-M cluster in a high availability configuration and as a replicated entity in disaster recovery configurations. Note that you need to power off virtual machines before attaching or detaching storage.

To attach storage to virtual machines:

1. Power off the virtual machine.
2. Log in to the RHEV-M machine.
3. Run `# /opt/VRTSrhevm/bin/vxrhevadm -p` *`RHEV Admin Password`* `-n` *`[VM] [dmpnodepath]`* `attach`

   Attaches the DMP node as a disk to the specified virtual machine, where *dmpnodepath* is the path of DMP device on the host.
4. Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <`*`password`*`> -n` *`[VM] [volume]`* `attach`

   Attaches *volume* as a block device to the specified virtual machine.

5 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> -n [VM] \ <file-path:raw> | <file-path:qcow2> attach`

Attaches file system as a file device to the specified virtual machine.

6 Power on the virtual machine either through the `vxrhevadm` utility or the RHEV-M web interface.

To detach storage to a virtual machine:

1 Power off the virtual machine.

2 Log in to the RHEV-M machine.

3 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> -n [VM] [dmpnodepath] detach`

Detaches the DMP node as a disk to the specified virtual machine, where *dmpnodepath* is the path of DMP device on the host.

4 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> -n [VM] [volume] detach`

Detaches the *volume* as a block device to the specified virtual machine.

5 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> -n [VM] \ <file-path:raw> | <file-path:qcow2> detach`

Detaches the filesystem as a file device to the specified virtual machine.

6 Power on the virtual machine either through the `vxrhevadm` utility or the RHEV-M web interface.

Listing configuration details of virtual machines in an RHEV environment

The List command lets you view the virtual machines and attached devices on a particular host.

To view the virtual machine details on a host:

1 Log in to the RHEV-M machine.

2 Run the `vxrhevdm` utility.

3 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> list hosts`

Lists the host address and host id.

4 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> list vms`

Lists the virtual machines present on the host and the associated Storage Foundation components.

5 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> list devices`

Lists the devices attached to virtual machines on the host.

Configuring permissions for exported storage devices

The configure command sets necessary permissions on all attached devices present on the specified host. When a host restarts or goes offline for some reason, the permissions of attached devices are reset. To run virtual machine with attached devices or to migrate virtual machine to another host, reset permissions for these attached devices.

Permissions on attached devices are reset because of following reasons:

1. Host is restarted
2. Disk group is deported and imported
3. Veritas Volume Manager (VxVM) is updated

To configure storage devices from the RHEV-M machine:

- Run `# /opt/VRTSrhevm/bin/vxrhevadm -p <password> -h [host] configure`

  Set necessary permissions for all the attached devices on the specified host.

Starting and stopping virtual machines

Perform start and stop operations from the vxrhevadm utility to start or stop virtual machines.

To start and stop a virtual machine:

1 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p admin -n [VM] start`

Starts the specified virtual machines.

2 Run `# /opt/VRTSrhevm/bin/vxrhevadm -p admin -n [VM] stop`

Stops the specified virtual machines.

Usage examples from the RHEV administration utility

A few command usages from the RHEV administration utility, `vxrhevadm`, to configure storage for virtual machines.

```
# ./vxrhevadm -p admin list hosts
```

Lists details of the virtual machines present on the host.

```
Host                Address                                  HostID
================================================================================
linux          veritas.com          e20b2608-1472-4f20-b54f-ae13af4907d4


# ./vxrhevadm -p admin list vms

VM                  State      Host       VMID
================================================================================
VM1                down         -          2d7f3d3d-adf2-4c70-a138-c646c7e6d771
VM2                down         -          813e5b85-8544-4fb9-a238-6c650cd73e49
VM3                down         -          af3d99e6-d007-4499-8d22-fc21e6f9f3d4
VM4                down         -          af62977b-8ba7-46b0-bca9-3828ca5354b9
VM5                down         -          308a4812-812d-44f5-9171-949025f38ef2
VM6                down         -          565ed6cd-d780-484b-84a6-1a5267a4eb72
VM7                down         -          e0141c6e-03d9-4eb0-8252-97dee1ba0a85
VM8                down         -          d3349764-49d6-4d2a-b8af-eb4068d61379

# ./vxrhevadm -p admin -n VM4 start
The virtual machine VM4 is started successfully.

# ./vxrhevadm -p admin -n VM4 start
The virtual machine VM4 is started successfully.


 # ./vxrhevadm -p admin -n VM4 -d /dev/vx/dmp/xiv0_8275 attach

Power off the virtual machine before you attach  the device.

 # ./vxrhevadm -p admin -n VM4 stop

The virtual machine VM4 is stopped successfully.
```

```
# ./vxrhevadm -p admin -n VM4 -d /dev/vx/dmp/xiv0_8274 attach

The device /dev/vx/dmp/xiv0_8274 is successfully attached to the virtual machine VM4.


 # ./vxrhevadm -p admin -n VM4 -d /dev/vx/dmp/xiv0_8274 detach

The device /dev/vx/dmp/xiv0_8274 is successfully detached from the virtual machine VM4.


# ./vxrhevadm -p admin -n VM5 -v  /dev/vx/dsk/group/vol3 detach

The device /dev/vx/dsk/group/vol3 is successfully detached from the virtual machine VM5.


 # ./vxrhevadm -p admin -n VM5 -f /mnt/disk.img:raw attach
The device /mnt/disk.img:raw is successfully attached to the virtual machine VM5.


 # ./vxrhevadm -p admin -h linux configure
The permissions of all attached devices present on host Linux are successfully configured.
```

Mapping DMP meta-devices

Consistent mapping can be achieved from the host to the guest by using the Persistent Naming feature of DMP.

Running DMP in the host has other practical benefits:

- Multi-path device can be exported as a single device. This makes managing mapping easier, and helps alleviate the 32 device limit, imposed by the VirtIO driver.
- Path failover can be managed efficiently in the host, taking full advantage of the Event Source daemon to proactively monitor paths.
- When Veritas InfoScale Solutions products are installed in the guest, the 'Persistent Naming' feature provides consistent naming of supported devices from the guest through the host to the array. The User Defined Names feature, or UDN, allows DMP virtual devices to have custom assigned names.

To map a DMP meta-device to a guest

- Map the device to the guest. In this example the dmp device *xiv0_8614* is mapped to *guest_1*.

  ```
  # # /opt/VRTSrhevm/bin/vxrhevadm  -p RHEV-M Admin Password -n guest_1  -d
  ```

  Where *RHEV-M Admin Password* is the administrator password on RHEV Manager.

  As DMP devices are attached through virtio-scsi interface, the SCSI commands are directly passed to the device which makes SCSI inquiry possible in the guest resulting into correct device identification. For volume and file devices, guest to host device mapping is not possible.

Note: Currently, you cannot identify volume and file system mappings to SCSI disks in the guest. You may run heuristics to identify device mappings in the guest.

Resizing devices

Red Hat Linux Enterprise (RHEL) 6.3, 6.4, and 6.5 do not support online disk re-sizing of VirtIO devices. To re-size a VirtIO device, the guest must be fully shut down and re-started.

You can use the following methods to resize the devices.

To grow devices

1. Grow the storage.
   - If the storage device is a VxVM Volume, re-size the volume.

     ```
     # vxassist -g <diskgroup> growto <volume> <new_len>
     ```

   - If the storage device is a LUN from a storage array, re-size the device on the array.
2. Update the size of the disk device in the host.
   - Stop all virtual machines using the storage device.
   - If the device is a LUN from a storage array, update the size of the device:

     ```
     # blockdev --rereadpt device
     ```

   - Restart the virtual machines.
3. Update the size of the storage device in the guest .

- If VxVM is managing the storage in the guest, use the `vxdisk resize` command.
- If VxVM is not managing the storage in the guest, see the appropriate documentation.

To shrink devices

1. Update the size of the disk device in the guest.
   - If VxVM is managing the device in the guest, if necessary, first use the vxresize utility to shrink any file systems and volumes which are using the device. Use the vxresize utility to update the size of the public region of the device:

     ```
     # vxdisk resize access_name length=new_size
     ```

   - If VxVM is not managing the storage in the guest, see the appropriate documentation.
2. Shrink the storage in the guest.
   - If the device is a VxVM volume, shrink the volume with the vxassist utility.
   - If the device is a LUN from a storage array, shrink the device on storage array.
3. Update the size of the disk device in the host.
   - Stop the guests which are using the devices.
   - If the device is a LUN from a storage array, use the following command:

     ```
     # blockdev --rereadpt device
     ```

4. Start the guests.

Section 4

Implementing Linux virtualization use cases

Chapter 7

Application visibility and device discovery

This chapter includes the following topics:

About storage to application visibility using

Datacenters adopt virtualization technology to effectively use the IT-infrastructure and substantially reduce the capital and operational expenditures. If you have adopted virtualization technology in your datacenter, provides you an efficient way of discovering and managing your virtual storage and infrastructure assets.

In your datacenter, helps you view the following relationships:

- Applications in your datacenter that manages and the virtual hosts on which they are running.
- Physical storage in your datacenter that is exported to the virtual machines.
- Physical storage in your datacenter that is exported to the virtual machines.

supports the following virtualization technologies:

- VMware
- Microsoft Hyper-V
- Kernel-based Virtual Machine (KVM)
- RedHat Enterprise Virtualization (RHEV)

In the VMware virtualization technology, a designated Control Host discovers the VMware vCenter Server in the datacenter. This discovery displays those ESXi servers that VMware vCenter Server manages, and the virtual machines that are configured on the ESXi servers.

For more information, see the *Veritas InfoScale™ Solutions Virtualization Guide for Linux on ESXi*

For Microsoft Hyper-V, discovers Hyper-V virtual machines and their correlation with the Hyper-V server. It also discovers the storage that is provisioned to the guests, and its correlation with the virtual machine and Hyper-V server. The Hyper-V guest (with or without `VRTSsfmh` RPM), when added to Management Server domain, provides storage mapping discovery.

For Kernel-based Virtual Machine (KVM), discovers KVM virtual machines on the Linux host if the KVM modules are installed, and configured. discovers basic information about only running virtual machines. For example, virtual machine name, CPU, and so on.

See "About Microsoft Hyper-V virtualization discovery" on page 105.

For more information, see the documentation.

About Kernel-based Virtual Machine (KVM) virtualization discovery in Veritas InfoScale Operations Manager

Kernel-based Virtual Machine (KVM) is a full virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V). Veritas InfoScale Operations Manager discovers KVM virtual machines on the Linux host if the KVM modules are installed, and configured. Veritas InfoScale Operations Manager discovers basic information about only running virtual machines. For example, virtual machine name, CPU, and so on. Veritas InfoScale Operations Manager uses `virsh` commands to discover KVM-related information.

Kernel-based Virtual Machine (KVM) discovery pre-requisites are as follows:

- `VRTSsfmh` package must be present on the Linux host.
- KVM modules must be installed and configured.

Kernel-based Virtual Machine (KVM) discovery limitations are as follows:

- Veritas InfoScale Operations Manager discovers only running virtual machines.
- Exported storage discovery, and storage correlation is not supported.

About Red Hat Enterprise Virtualization (RHEV) virtualization discovery in Veritas InfoScale Operations Manager

Red Hat Enterprise Virtualization (RHEV) is a desktop and server virtualization platform based on the KVM hypervisor as well as the Red Hat Enterprise Linux (RHEL) server operating system. It provides a RHEL-based centralized management server, RHEV-M, with a web-based interface for managing virtual machines (VMs). RHEV uses SPICE protocol and Virtual Desktop Server Manager (VDSM) along with RHEV-M.

Veritas InfoScale Operations Manager discovers RHEV virtual machines on the Linux host if the RHEV modules are installed, and configured. Veritas InfoScale Operations Manager discovers basic information about only running virtual machines.

RHEV-based Virtual Machine (RHEV) discovery pre-requisites are as follows:

- `VRTSsfmh` package must be present on the Linux host.
- RHEV modules must be installed and configured.

RHEV-based Virtual Machine (RHEV) discovery limitations are as follows:

- Veritas InfoScale Operations Manager discovers only running virtual machines.
- Exported storage discovery, and storage correlation is not supported.

About Microsoft Hyper-V virtualization discovery

Hyper-V is a hypervisor-based virtualization technology from Microsoft for x86-64 systems. You can use Veritas InfoScale Operations Manager to discover Hyper-V host and virtual machine-related information if the Hyper-V role is enabled on the managed host. Veritas InfoScale Operations Manager uses the Hyper-V WMI API for the discovery.

Hyper-V discovery can be grouped into the following categories:

- Virtual machine discovery: Hyper-V virtual machine discovery by Veritas InfoScale Operations Manager and its correlation with the Hyper-V server.

- Exported storage discovery: Discovery of storage that is provisioned to the guests and its correlation with the virtual machine and Hyper-V server.

See "Virtual machine discovery in Microsoft Hyper-V " on page 106.

See "Storage mapping discovery in Microsoft Hyper-V" on page 106.

Virtual machine discovery in Microsoft Hyper-V

Veritas InfoScale Operations Manager lets you discover information about Hyper-V virtual machines. For example, the name of the virtual machine, allocated memory, CPU, state, and the storage exported (virtual hard disks and pass through disks) from Hyper-V server to Hyper-V guest. Veritas InfoScale Operations Manager discovers all virtual machines including the virtual machines without the guest operating system installed.

Agent and agentless discoveries of Hyper-V virtual machines are supported. However, for the agentless method, the discovered information is limited. To discover more information about the configured virtual machines, the agent discovery method should be used. It provides detailed information about the virtual machines.

For more information on agent and agentless discovery, see the *Veritas Operations Manager Management Server Administrator's Guide*

Virtual machine discovery prerequisites are as follows:

- The `VRTSsfmh` package should be installed on the Hyper-V server (parent partition).
- The Hyper-V role should be enabled.
- The Windows Management Instrumentation (WMI) service should be running.

A limitation of virtual machine discovery is listed below:

- Hyper-V discovery is not supported on an agentless Hyper-V Server (parent partition) to which the Hyper-V virtual machines are associated.

See "About Microsoft Hyper-V virtualization discovery" on page 105.

Storage mapping discovery in Microsoft Hyper-V

Veritas InfoScale Operations Manager discovers the storage provisioned to the guests from the host's local storage, or storage area network (SAN). The Hyper-V guest (with or without `VRTSsfmh` package), when added to the Veritas InfoScale Operations Manager Management Server domain, provides storage mapping discovery.

Additional storage attributes are also displayed on the page. For example, size, type of the storage (VHD or passthrough disk), and the storage container (volume on the host where virtual storage is provisioned). The storage device handles on the guest will be mapped to the corresponding VHD or passthrough disk provisioned from host. Veritas InfoScale Operations Manager also discovers the snapshot disks provisioned to the VMS.

The storage mapping discovery prerequisites are as follows:

- The Hyper-V server must be running Microsoft Windows 2008 R2 or later operating system.
- Windows Management Instrumentation (WMI) should be running on the guest.

The storage mapping discovery limitation is as follows:

- Storage correlation is not supported for Linux guests.

For more information on storage mapping and storage correlation, see the *Veritas Operations Manager Management Server Administrator's Guide*.

See “About Microsoft Hyper-V virtualization discovery” on page 105.

Chapter 8

Server consolidation

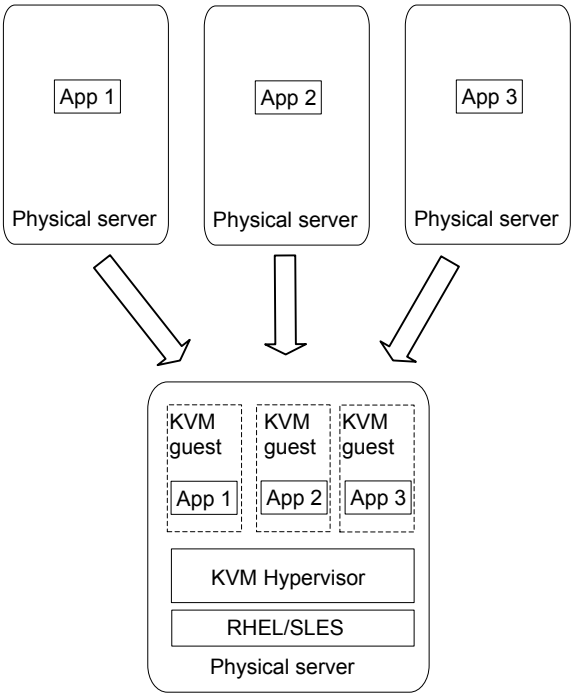
This chapter includes the following topics:

Server consolidation

Storage Foundation and High Availability Solutions products can be used in many combinations. The configurations listed are the minimum required to accomplish the objectives of the respective use cases.

Server consolidation enables you to run multiple virtual machines, each with the functionality equivalent to a physical server, combining the multiple applications and their workloads onto a single server for better server utilization and reduced datacenter server sprawl.

Figure 8-1 Server consolidation



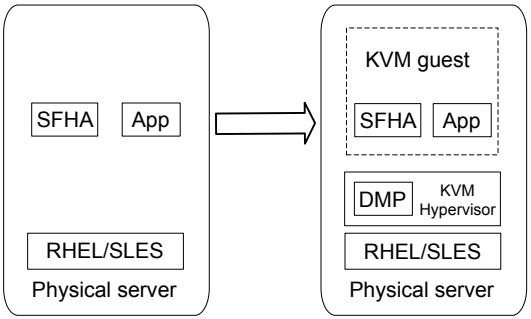
The server consolidation use case is supported for the following Linux virtualization technologies:

- Red Hat Enterprise Linux (RHEL) KVM
- SUSE Linux Enterprise Server (SLES) KVM
- Red Hat Enterprise Virtualization (RHEV)

Implementing server consolidation for a simple workload

This solution for a single server with Storage Foundation HA illustrates the migration of a single workload into a KVM Guest.

Figure 8-2 Server consolidation for a simple workload



To implement server consolidation for a simple workload

1. Install Veritas InfoScale Enterprise product and configure SFHA in the virtual machine.

   See "Installing Veritas InfoScale Solutions in the kernel-based virtual machine environment" on page 45.

2. Map the storage from the array to the host.
3. Map the storage from the array to the guest.

   See "Mapping devices to the guest" on page 50.

4. Go into the guest and make sure you can import disk groups.

Chapter 9

Physical to virtual migration

This chapter includes the following topics:

Physical to virtual migration

Migrating data from physical servers to virtual machines can be painful. Veritas InfoScale Solutions products can make painful migrations of data from physical to virtual environments easier and safer to execute.

With Veritas InfoScale Solutions, there is no need to copy any data from source to destination, but rather the administrator reassigns the same storage or a copy of the storage for a test migration, to the virtual environment. Data migration with Storage Foundation (SF), Storage Foundation HA (SFHA), or Storage Foundation Cluster File System High Availability (SFCFSHA) can be executed in a central location, migrating all storage from an array utilized by Storage Foundation managed hosts.

Physical to virtual migration (P2V) requires migrating data from a physical server to a virtualized guest. The LUNs are first physically connected to the host, and then the LUNs are mapped in KVM from the host to the guest.

Without SF, SFHA, or SFCFSHA in the host, you must identify which storage devices with mapping to the guest. Putting SF, SFHA, or SFCFSHA in the host enables quick and reliable identification of storage devices to be mapped. If you are running DMP in the host, you can map the DMP devices directly. Veritas InfoScale Solutions products add manageability and ease of use to an otherwise tedious and time-consuming process.

The physical to virtual migration use case is supported for the following Linux virtualization technologies:

- Red Hat Enterprise Linux (RHEL) KVM
- SUSE Linux Enterprise Server (SLES) KVM
- Red Hat Enterprise Virtualization (RHEV)

How to implement physical to virtual migration (P2V)

Migrating data from a physical server to a virtualized guest, the LUNs are first physically connected to the host, and then the LUNs are mapped in KVM from the host to the guest.

This use case procedure is very similar to the server consolidation use case and the procedures are quite similar. Physical to virtual migration is the process used to achieve server consolidation.

This use case requires Storage Foundation HA or Storage Foundation Cluster File System HA in the KVM host and Storage Foundation in the KVM guest. For setup information:

See "Installing Veritas InfoScale Solutions in the kernel-based virtual machine environment" on page 45.

There are three options:

- If Veritas InfoScale Solutions products are installed on both the physical server and the virtual host, identifying the LUNs which need mapping is made easy. Once the LUNs are connected to the virtual host, 'vxdisk –o alldgs list' can be used to identify the devices in the disk group which require mapping.
- If Veritas InfoScale Solutions products are not installed on the virtual host and the physical server is a Linux system, the devices which need mapping can be identified by using the device IDs on the physical server.
- If Veritas InfoScale Solutions products are installed only on the physical server and the SF administration utility for RHEV, `vxrhevadm`, is installed on the RHEV-M machine, you can identify the exact DMP device mapping on the guest. However, for volume and file system mappings, run heuristics to identify exact device mappings on the host.

To implement physical to virtual migration with Storage Foundation in the host and guest (KVM-only)

1 Find the Linux device IDs of the devices which need mapping.

```
# vxdg list diskgroup
```

2 For each disk in the disk group:

```
# vxdmpadm getsubpaths dmpnodename=device
# ls -al /dev/disk/by-id/* | grep subpath
```

If Storage Foundation is not installed on the host, before decommissioning the physical server, identify the LUNs which require mapping by using the devices serial numbers. The LUNs can be mapped to the guest using the persistent "by-path" device links.

To implement physical to virtual migration if Storage Foundation is not installed in the host (KVM-only)

1. On the physical server, identify the LUNs which must be mapped on the KVM host using the `udevadm` command.
2. Map the LUNs to the virtualization host.

   The udev database can be used to identify the devices on the host which need to be mapped.

   ```
   # udevadm info --export-db | grep '/dev/disk/by-path' | \
                   cut -d' ' -f4

   /dev/disk/by-path/pci-0000:05:00.0-fc-0x5006016239a01884-lun-1
   /dev/disk/by-path/pci-0000:05:00.0-fc-0x5006016239a01884-lun-2
   ```

   Map the LUNs to the guest. As there are multiple paths in this example, the paths sym-link can be used to ensure consistent device mapping for all four paths.

   ```
   # virsh attach-disk guest1 \
   /dev/disk/by-path/pci-0000:05:00.0-fc-0x5006016239a01884-lun-1 \
         vdb
   # virsh attach-disk guest1 \
   /dev/disk/by-path/pci-0000:05:00.0-fc-0x5006016239a01884-lun-2 \
         vdc
   ```

3. Verify that the devices are correctly mapped to the guest. The configuration changes can be made persistent by redefining the guest.

   ```
   # virsh dumpxml guest1 > /tmp/guest1.xml
   # virsh define /tmp/guest1.xm
   ```

To implement physical to virtual migration with Storage Foundation in the guest and host (KVM-only)

1. Map the LUNs to the virtualization host.
2. On the virtualization host, identify the devices which require mapping. For example, the devices with the disk group *data_dg* are mapped to *guest1*.

   ```
   # vxdisk -o alldgs list |grep data_dg
   3pardata0_1  auto:cdsdisk    -            (data_dg)    online
   3pardata0_2  auto:cdsdisk    -            (data_dg)    online
   ```

3 Map the devices to the guest.

```
# virsh attach-disk guest1 /dev/vx/dmp/3pardata0_1 vdb
Disk attached successfully

# virsh attach-disk guest1 /dev/vx/dmp/3pardata0_2 vdc
Disk attached successfully
```

4 In the guest, verify that all devices are correctly mapped and that the disk group is available.

```
# vxdisk scandisks
# vxdisk -o alldgs list |grep data_dg
3pardata0_1  auto:cdsdisk    -            (data_dg)    online
3pardata0_2  auto:cdsdisk    -            (data_dg)    online
```

5 In the virtualization host make the mapping persistent by redefining the guest:

```
# virsh dumpxml guest1 > /tmp/guest1.xml
# virsh define /tmp/guest1.xml
```

To implement physical to virtual migration with Storage Foundation only in the guest and the SF administration utility for RHEV, `vxrhevadm`, on the RHEV Manager

1 Map the LUNs to the virtualization host.

2 On the virtualization host, identify the devices which require mapping. For example, the devices with the disk group *data_dg* are mapped to *guest1*.

```
# vxdisk list -guest1 <data_dg> DMP nodes
# vxprint  -guest1 <data_dg> -v, volume
# vxfs,  file created on vxfs filesystem
```

3 2. Attach each entity to respective virtual machines.

```
# ./vxrhevadm -p <password> -n <VM name> -d <dmpnode> attach
Attached a dmp node to the specified virtual machine

#  ./vxrhevadm -p <password> -n <VM name> -v <volume> attach
Attached a volume device to the specified virtual machine

# ./vxrhevadm  -p <password> -n <VM name> -f <file>:raw attach
Attached a file system  device to the specified virtual machine
```

4 Power up the guest virtual machine and verify that the SCSI disks are available in the guest virtual machine.

Note: The XML dumps available in the `/var/log/vdsm/vdsm.log` is a hint about device mappings. For DMP nodes, enable persistent naming in the host to identify the device mapping in the guest. For volume and file system mappings, run heuristics to identify device mappings in the guest.

To use a Veritas Volume Manager volume as a boot device when configuring a new virtual machine

1 Follow the recommended steps in your Linux virtualization documentation to install and boot a VM guest.

When requested to select managed or existing storage for the boot device, use the full path to the VxVM storage volume block device, for example */dev/vx/dsk/boot_dg/bootdisk-vol*.

2 If using the `virt-install` utility, enter the full path to the VxVM volume block device with the --disk parameter, for example, *--disk path=/dev/vx/dsk/boot_dg/bootdisk-vol*.

To use a Storage Foundation component as a boot device when configuring a new virtual machine

1. Follow the recommended steps in your Linux virtualization documentation to install and boot a VM guest.

   When requested to select managed or existing storage for the boot device, use the full path to the VxVM storage volume block device, file system device, or DMP node.

   For example */dev/vx/dsk/boot_dg/bootdisk-vol*

   Likewise, */dev/vx/dsk/boot_dg/bootdisk-file*, or */dev/vx/dsk/boot_dg/bootdisk-dmpnode*.

2. In the RHEV Manager advanced settings for virtual machines, select the boot option and attach the appropriate ISO image.

3. Attach the DMP node, volume block device, or file system device as the boot option.

   ```
   # /opt/VRTSrhevm/bin/vxrhevadm -p \
   <rhevm-password> -n <vmname> -d <dmpnode-path> attach
   # /opt/VRTSrhevm/bin/vxrhevadm -p \
   <rhevm-password> -n <vmname> -v <volume-path> attach
   # /opt/VRTSrhevm/bin/vxrhevadm -p \
   <rhevm-password> -n <vmname> -f <file-path:raw> |
   <file-path:qcow2> attach
   ```

4. Start the guest virtual machine and boot from ISO.

5. Install OS on the SF entity appearing as a SCSI device. Install bootloader on the SCSI device itself.

6. Power off the guest virtual machine.

7. Configure the host to boot from hard disk in guest virtual machine settings.

8. Power on the guest to book from the configured SF component.

Chapter 10

Simplified management

This chapter includes the following topics:

Simplified management

Independent of how an operating system is hosted, consistent storage management tools save an administrator time and reduce the complexity of the environment. Veritas InfoScale Solutions products in the guest provide the same command set, storage namespace, and environment as in a non-virtual environment.

This use case requires Storage Foundation HA or Storage Foundation Cluster File System HA in the KVM or RHEV host. For setup information:

See "Installing Veritas InfoScale Solutions in the kernel-based virtual machine environment" on page 45.

The simplified management use case is supported for the following Linux virtualization technologies:

- Red Hat Enterprise Linux (RHEL) KVM
- SUSE Linux Enterprise Server (SLES) KVM
- Red Hat Enterprise Virtualization (RHEV)

Provisioning storage for a guest virtual machine

A volume can be provisioned within a VM guest as a data disk or a boot disk.

- Data disk: provides the advantage of mirroring data across arrays.

- Boot disk: provides the ability to migrate across arrays.

Adding a VxVM storage volume as a data disk to a running guest virtual machine can be done in the following ways:

- Using the `virt-manager` console.
- Using the `virsh` command line.
- Using the `VRTSrhevm` utility or RHEV-M web interface.

Provisioning Veritas Volume Manager volumes as data disks for VM guests

The following procedure uses Veritas Volume Manager (VxVM) volumes as data disks (virtual disks) for VM guests. The example host is *sys1* and the VM guest is *guest1*. The prompts in each step show in which domain to run the command.

To provision Veritas Volume Manager volumes as data disks

1. Create a VxVM disk group (*mydatadg* in this example) with some disks allocated to it:

   ```
   sys1# vxdg init mydatadg TagmaStore-USP0_29 TagmaStore-USP0_30
   ```

2. Create a VxVM volume of the desired layout (in this example, creating a simple volume):

   ```
   sys1# vxassist -g mydatadg make datavol1 500m
   ```

3. KVM: Map the volume *datavol1* to the VM guest:

   ```
   sys1# virsh attach-disk guest1 /dev/vx/dsk/mydatadg/datavol1 vdb
   ```

   RHEV: Map the volume *datavol1* to the VM guest:

   ```
   #  /opt/VRTSrhevm/bin/vxrhevadm -p  RHEV Admin Password -n <VM> -v <volum
   ```

4. To make the mapping persistent, redefine the VM guest.

   KVM:

   ```
   sys1# virsh dumpxml guest1 > /tmp/guest1.xml
   ```

   ```
   sys1# virsh define /tmp/guest1.xml
   ```

Provisioning Veritas Volume Manager volumes as boot disks for guest virtual machines

The following procedure outlines how to provision a Veritas Volume Manager (VxVM) volume as a boot disk for guest virtual machines.

The example host is *sys1* the VM guest is *guest1*. The prompts in each step show in which domain to run the command.

To provision Veritas Volume Manager volumes as boot disks for guest virtual machines

1. On the host, create a VxVM volume. Use the size that is recommended by your Linux documentation. In this example, a 16GB volume is created:

   ```
   sys1# vxassist -g boot_dg make bootdisk-vol 16g
   ```

2. Follow the recommended steps in your Linux documentation to install and boot a VM guest, and use the virtual disk as the boot disk.

Boot image management

With the ever-growing application workload needs of datacenters comes the requirement to dynamically create virtual environments. This creates a need for the ability to provision and customize virtual machines on-the-fly. Every virtual machine created needs to be provisioned with a CPU, memory, network and I/O resources.

As the number of guest virtual machines increase on the physical host, it becomes increasingly important to have an automatic, space-optimizing provisioning mechanism. Space-savings can be achieved as all the guest virtual machines can be installed with the same operating system, i.e., boot volume. Hence, rather than allocate a full boot volume for each guest, it is sufficient to create single boot volume and use space-optimized snapshots of that “Golden Boot Volume” as boot images for other virtual machines.

The primary I/O resource needed is a boot image, which is an operating system environment that consists of: the following

- A bootable virtual disk with the guest operating system installed
- A bootable, a guest file system
- A custom or generic software stack

For boot image management, Veritas InfoScale Solutions products enable you to manage and instantly deploy virtual machines based on templates and snapshot-based boot images (snapshots may be full or space optimized). For

effective boot image management in KVM based virtual environments, deploy the Veritas InfoScale Solutions products in the combined host and guest configuration.

Benefits of boot image management:

- Eliminates the installation, configuration and maintenance costs associated with installing the operating system and complex stacks of software
- Infrastructure cost savings due to increased efficiency and reduced operational costs.
- Reduced storage space costs due to shared master or gold image as well as space-optimized boot images for the various virtual machines
- Enables high availability of individual guest machines with Cluster Server (running on the host) monitoring the VM guests and their boot images
- Ability to create and deploy virtual machines across any remote node in the cluster

Creating the boot disk group

Once Storage Foundation HA (SFHA) is configured on the Linux server using the combined host and VM guest configuration, the next step is to create a disk-group in which the Golden Boot Volume and all the various space-optimized snapshots (VM boot images) will reside. For a single-node environment, the disk-group is local or private to the host. For a clustered environment (recommended for live migration of VMs), Veritas recommends creating a shared disk-group so that the Golden Boot Volume can be shared across multiple physical nodes.

It is possible to monitor the disk-group containing the Guest VM boot image(s) and the guest VMs themselves under VCS so that they can be monitored for any faults. However it must be kept in mind that since the boot images are in the same disk-group, a fault in any one of the disks backing the snapshot volumes containing the boot disks can cause all the guest VMs housed on this node to failover to another physical server in the Storage Foundation Cluster File System High Availability (SFCFS HA) cluster. To increase the fault tolerance for this disk-group, mirror all volumes across multiple enclosures making the volumes redundant and less susceptible to disk errors.

To create a shared boot disk group

1. Create a disk group, for example *boot_dg*.

```
$ vxdg -s init boot_dg  device_name_1
```

2. Repeat to add multiple devices.

```
$ vxdg -g boot_dg adddisk device_name_2
```

Creating and configuring the golden image

The basic idea is to create a point-in-time image based on a master or gold image. The image will serve as the basis for all boot images once it is set up. Hence, first set up a complete virtual machine boot volume as a golden boot volume.

To create the golden image

1. In the selected disk group, create a VxVM volume. Use the size that is recommended by your Linux documentation. For example, the disk group is *boot_dg*, the golden boot volume is *gold-boot-disk-vol*, the volume size is 16GB.

   ```
   sys1# vxassist -g boot_dg make gold-boot-disk-vol 16g
   ```

2. Follow the recommended steps in your Linux documentation to install and boot a VM guest.

   When requested to select managed or existing storage for the boot device, use the full path to the VxVM storage volume block device.

   For example: */dev/vx/dsk/boot_dg/gold-boot-disk-vol*.

3. If using the `virt-install` utility, enter the full path to the VxVM volume block device with the --disk parameter.

   For example: *--disk path=/dev/vx/dsk/boot_dg/gold-boot-disk-vol*.

4. After the virtual machine is created, install any guest operating system with the boot volume and the virtual machine configured exactly as required.

5. After the virtual machine is created and configured, shut it down.

You can now use the boot image as a image (hence called a golden image) for provisioning additional virtual machines that are based on snapshots of the Golden Boot Volume. These snapshots can be full copies (mirror images) or they can be space-optimized snapshots. Using space-optimized snapshots greatly reduces the storage required to host the boot disks of identical multiple virtual machines. Note that since both, the full and space-optimized snapshots, are instantly available (no need to wait for the disk copy operation), provisioning of new virtual machines can now be instantaneous as well.

Rapid Provisioning of virtual machines using the golden image

As mentioned above, for rapid provisioning of new virtual machines based on the golden image, we need to have full or space-optimized snapshots of the Golden Boot Volume. These snapshots can then be used as boot images for the new virtual machines. The process to create these snapshots is outlined below in the procedures below.

Creating Instant, Full Snapshots of Golden Boot Volume for Rapid Virtual Machine Provisioning

To create instant, full snapshots of the golden boot volume for rapid virtual machine provisioning

1. Prepare the volume for an instant full snapshot. In the example, the disk group is *boot_dg* and the golden boot volume is *gold-boot-disk-vol*.

   ```
   $ vxsnap -g boot_dg prepare gold-boot-disk-vol
   ```

2. Create a new volume which will be used as the boot volume for the new provisioned guest. The size of the guests boot volume must match the size of the golden boot volume.

   ```
   $ vxassist -g boot_dg make guest1-boot-disk-vol 16g layout=mirror
   ```

3. Prepare the new boot volume so it can be used as a snapshot volume.

   ```
   $ vxsnap -g boot_dg prepare guest1-boot-disk-vol
   ```

4. Create the full instant snapshot of the golden boot volume.

   ```
   $ vxsnap -g boot_dg make source=gold-boot-disk-vol/snapvol=\
     guest1-boot-disk-vol/syncing=off
   ```

5. Create a new virtual machine, using the snapshot *guest1-boot-disk-vol* as an "existing disk image."

To create instant, space-optimized snapshots of the golden boot volume for rapid virtual machine provisioning

1. Prepare the volume for an instant snapshot. In the example, the disk group is*boot_dg* and the golden boot volume is *gold-boot-disk-vol*.

   ```
   $ vxsnap -g boot_dg prepare gold-boot-disk-vol
   ```

2. Use the `vxassist` command to create the volume that is to be used for the cache volume. The cache volume will be used to store writes made to the space-optimized instant snapshots.

   ```
   $ vxassist -g boot_dg make cache_vol 5g  layout=mirror init=active
   ```

3. Use the `vxmake cache` command to create a cache object on top of the cache volume which you created in the previous step.

   ```
   $ vxmake -g boot_dg cache cache_obj cachevolname=cache_vol autogrow=on
   ```

4 Start the cache object:

```
$ vxcache -g boot_dg start cache_obj
```

5 Create a space-optimized instant snapshot of the golden boot image:

```
$ vxsnap -g boot_dg make source=\
gold-boot-disk-vol/newvol=guest1-boot-disk-vol/cache=cache_obj
```

6 Create a new virtual machine, using the snapshot of the golden image as an existing disk image.

Storage Savings from space-optimized snapshots

With the large number of virtual machines housed per physical server, the number of boot images used on a single server is also significant. A single bare-metal Linux boot image needs around 3 GB of space at a minimum. Installing software stacks and application binaries on top of that requires additional space typically resulting in using around 6 GB of space for each virtual machine that houses a database application.

When a user provisions a new virtual machine, the boot image can be a full copy or a space-optimized snapshot. Using a full copy results in highly inefficient use of storage. Not only is storage consumed to house identical boot images, storage is also consumed in making the boot images highly available (mirror across enclosures) as well in their backup.This large amount of highly available, high performance storage is very expensive, and likely to eliminate the cost advantages that server virtualization would otherwise provide. To add to it, backup and recovery of such capacity is also an expensive task.

In order to address the above issue, Veritas recommends the use of space-optimized snapshots of the gold image as boot images of the various VM guests. Space-optimized snapshots do not make a full copy of the data in the gold image, rather they work on the copy-on-write principle where only the changed blocks are stored locally. This set of changed blocks is called a Cache Object and it is stored in a repository for all such space-optimized snapshots, called the Cache Object Store, which is backed by physical storage. The Cache Object offers a significant storage space reduction, typically occupying a 5-20% storage footprint, relative to the parent volume (the gold image volume in this case). The same Cache Object Store can be used to store changed blocks for multiple snapshot volumes.

Each Snapshot held in the Cache Object Store contains only changes made to the gold image to support that installation’s boot environment. Hence, to achieve the best possible storage reduction, install software on data disks rather than root file

systems and limit as many changes as possible to the gold image operating files (i.e., system, hosts, passwd, etc.).

Chapter 11

Application availability using Cluster Server

This chapter includes the following topics:

About application availability options

Veritas products can provide the ultimate levels of availability in your KVM environment. In a KVM environment, you can choose a different combination of High Availability solutions: ApplicationHA and Cluster Server (VCS).

ApplicationHA by itself provides application monitoring and restart capabilities while providing ultimate visibility and manageability through Veritas Operations Manager. When ApplicationHA is adopted together with Cluster Server in the host, the two solutions work together to ensure that the applications are monitored and restarted if needed, and virtual machines are restarted if application restarts are not effective. These two solutions work together to provide the ultimate level of availability in your KVM environment.

If your KVM environment requires the same level of application availability provided by a VCS cluster in a physical environment, you can choose to adopt Cluster Server

in the virtual machines. In this configuration, your application enjoys fast failover capability in a VCS cluster in the virtual machines.

Table 11-1 Comparison of availability options

Required availability level	Recommended solution	Supported virtualization option
Application monitoring and restart	ApplicationHA in the virtual machines	Red Hat Enterprise Linux (RHEL) KVM
Virtual machine monitoring and restart	VCS cluster in the host monitoring the virtual machines as a resource	Red Hat Enterprise Linux (RHEL) KVM Red Hat Enterprise Virtualization (RHEV) SUSE Linux Enterprise Server (SLES) KVM
Combined application and virtual machine availability	ApplicationHA in the virtual machine and VCS cluster in the host	Red Hat Enterprise Linux (RHEL) KVM
Application failover to standby node in cluster	VCS cluster in the virtual machines	Red Hat Enterprise Linux (RHEL) KVM SUSE Linux Enterprise Server (SLES) KVM Red Hat Enterprise Virtualization (RHEV) Microsoft Hyper-V Oracle Virtual Machine (OVM)

Note: For application high availability and failover capabilities the application data must be on the shared storage accessible to all the nodes of the VCS cluster.

For setup information for ApplicationHA or VCS:

See "Installing and configuring Cluster Server in a kernel-based virtual machine (KVM) environment" on page 47.

Note: You can also use the cluster functionality of Storage Foundation HA or Storage Foundation Cluster File System HA if you need storage management capabilities in addition to application availability for your KVM environment.

Cluster Server In a KVM Environment Architecture Summary

VCS in host architecture

- Manages multiple guest virtual machines as a single unit of control
- Provides automatic restart or fail-over of individual guest virtual machines in response to failures
- Provides Start / Stop / Monitor of individual guest virtual machines from a common console across the entire server pool using Veritas Operations Manager (VOM)

VCS in guest architecture

- Manages applications running in the guest virtual machine as a single unit of control
- Provides automatic restart or fail-over of individual applications to other guest virtual machine or physical machine.
- Provides Start / Stop / Monitor of individual applications from a common console across appropriate guest virtual machines in the cluster using Veritas Operations Manager (VOM)

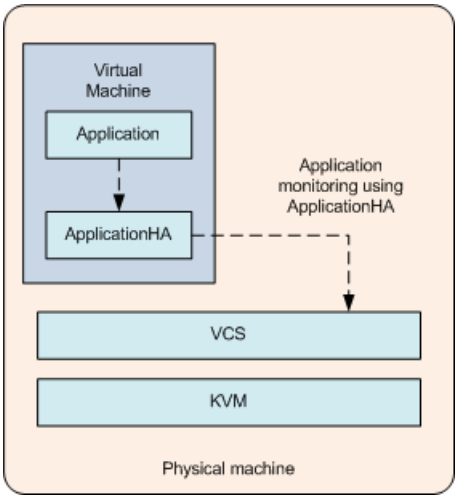
VCS in host to provide the Virtual Machine high availability and ApplicationHA in guest to provide application high availability

VCS running in the host monitors the virtual machine to provide the VM high availability. ApplicationHA running in the virtual machine (VM guest ensures the application high availability by monitoring the configured application. VCS and ApplicationHA can be combined together to provide the enhanced solution for achieving application and VM high availability.

VCS in host provides the primary VM monitoring. It can start/stop the virtual machine and fail-over it to another node in case of any fault. We then run ApplicationHA within the guest that monitors the application running inside the guest virtual machine. ApplicationHA in guest will not trigger an application fail-over in case of application fault, but it'll try to restart the application on same VM guest. If ApplicationHA fails to start the application, it can notify the VCS running in the host to take corrective action which includes virtual machine restart or virtual machine fail-over to another host.

For detailed information about ApplicationHA and integration of ApplicationHA with VCS, see the *ApplicationHA User's Guide*.

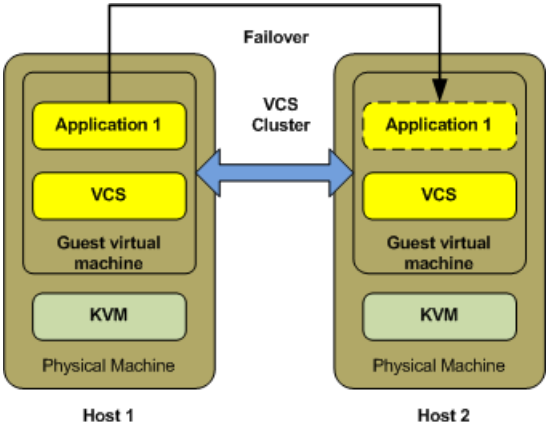
Figure 11-1 VCS In host for VM HA and ApplicationHA in guest for application HA



Virtual to Virtual clustering and failover

Running VCS in multiple guest virtual machines enables guest-to-guest clustering. VCS can then monitor individual applications running within the guest and then fail over the application to another guest in the virtual – virtual cluster.

Figure 11-2 Clustering between guests for application high availability

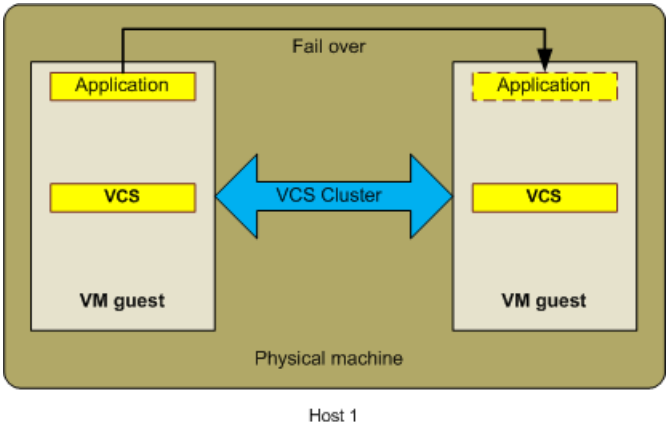


Note: I/O fencing support for clustering between guests for application high availability: SCSI3, Non-SCSI3, coordination point (CP) server based fencing is supported.

You can run VCS within each guest machine to provide high availability to applications running within the guest.

A VCS cluster is formed among the VM guests in this configuration. The VM guests in the cluster can be either on the same physical host or on different physical hosts. VCS is installed in the VM guests in the cluster. The VCS installation and configuration in a virtual machine is similar to that of VCS in the physical host clusters. This VCS cluster manages and controls the applications and services that run inside the VM guests. Any faulted application or service is failed over to other VM guest in the cluster. This configuration does not take care of the VM guest fail-overs since VCS runs inside the VM guest.

Figure 11-3 VCS cluster across VM guests on the same physical machine



Note: I/O fencing support for a VCS cluster across VM guests on the same physical machine: SCSI3, Non-SCSI3, CP server based fencing is supported.

I/O fencing support for Virtual to Virtual clustering

Disk-based SCSI-3 I/O fencing is supported for VCS clusters across VM guests on different physical machines. In case of VM as iSCSI initiator, virtual machines can be on same or different physical machines. The support matrix table lists the SCSI-3 fencing support for various Linux virtualization technologies.

Table 11-2 I/O fencing support matrix for virtual to virtual clustering

Virtualization Technology - Storage Protocol	SCSI Initiator	Guest OS	Supported(Yes/No)	Live Migration(Y/N)
KVM-iSCSI	VM as Initiator	RHEL 6.x	Y	Y
		SLES 11	Y	Y
		SLES 12	Y	Y
		RHEL 7.x		
KVM-iSCSI	Hypervisor as Initiator (Virtio-SCSI)	RHEL 6.x	Y	N
		SLES 11	N	N
		SLES 12	Y	N
		RHEL 7.x		
VMware-iSCSI	VM as Initiator	RHEL 6.x	Y	Y
		SLES 11	Y	Y
		SLES 12		
		RHEL 7.x		
VMware-iSCSI	Hypervisor as Initiator (RDMP)	RHEL 6.x	Y	N
		SLES 11	Y	N
		SLES 12		
		RHEL 7.x		
KVM-FCP	VM as Initiator (PCI - Passthrough)	RHEL 6.x	Y	N
		SLES 11	Y	N
		SLES 12	Y	N
		RHEL 7.x		
KVM-FCP	Hypervisor as Initiator (Virtio-SCSI)	RHEL 6.x	Y	N
		SLES 11	N	N
		SLES 12	Y	N
		RHEL 7.x		

Table 11-2 I/O fencing support matrix for virtual to virtual clustering *(continued)*

Virtualization Technology - Storage Protocol	SCSI Initiator	Guest OS	Supported(Yes/No)	Live Migration(Y/N)
VMware-FCP	VM as Initiator	RHEL 6.x	Y	N
		SLES 11	Y	N
		SLES 12		
		RHEL 7.x		
VMware-FCP	Hypervisor as Initiator (RDMP)	RHEL 6.x	Y	N
		SLES 11	Y	N
		SLES 12		
		RHEL 7.x		

Virtual to Physical clustering and failover

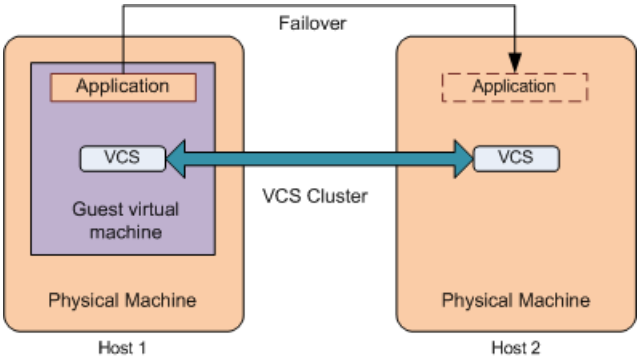
One can also create a physical to virtual cluster by combining VCS inside the virtual machine together with VCS running on any other physical host. This virtual-physical cluster enables VCS to monitor applications running within the guest and then fail over the application to another host. The reverse flow is also true, thus enabling the fail-over of an application running on a physical host into a VM guest machine.

A VCS cluster is formed among the VM guests and physical machines. VCS is installed on the VM guests and on different physical machines in the cluster. VM guests are connected to physical machines through the network of their VM hosts. In this case, the VM host is a physical machine on which one or more VM guests forming the cluster are hosted.

This VCS cluster manages and monitors the services and applications running on cluster nodes that can either be VM guests or physical machines. Any faulted application on one node fails over to other node that can either be a virtual machine or a physical machine.

See “Standard bridge configuration” on page 59.

Figure 11-4 VCS cluster across VM guest and physical machine



I/O fencing support: SCSI3, Non-SCSI3, CP server based fencing is supported.

Chapter 12

Virtual machine availability

This chapter includes the following topics:

About virtual machine availability options

While application availability is very important for KVM users, virtual machine availability is equally important. Virtual machine availability can be provided by adopting Cluster Server (VCS) in the host. VCS in this case monitors the virtual machines as a resource.

See Table 11-1 on page 127.

The virtual machine availability use case is supported for the following Linux virtualization technologies:

- Red Hat Enterprise Linux (RHEL) KVM
- Red Hat Enterprise Virtualization (RHEV)
- SUSE Linux Enterprise Server (SLES) KVM

For setup information for VCS for RHEL and SUSE:

See "Installing and configuring Cluster Server in a kernel-based virtual machine (KVM) environment" on page 47.

Note: For virtual machine high availability and failover capabilities the virtual machine image must be on the shared storage accessible to all the nodes of the VCS cluster.

Note: You can also use the cluster functionality of Storage Foundation HA or Storage Foundation Cluster File System HA if you need storage management capabilities in addition to virtual machine availability for your KVM host.

VCS in host monitoring the Virtual Machine as a resource

In this scenario, Cluster Server (VCS) runs in the host, enabling host-level clustering. Running VCS in the host also enables the monitoring and fail-over of individual guest virtual machines. Each guest virtual machine is simply a process in the KVM architecture and hence can be monitored by VCS running on the host. This capability allows us to monitor the individual virtual machine as an individual resource and restart/fail-over the VM on the same (or another physical) host. To enable support for guest live migration, Veritas recommends that you run Cluster Volume Manager (CVM) in the host.

In this configuration, the physical machines (PMs) hosting VM guests form a cluster. Therefore, VCS does not monitor applications running inside the guest virtual machines. VCS controls and manages the virtual machines with the help of the KVMGuest agent. If a VM guest faults, it fails over to the other host.

Note: The VM guests configured as failover service groups in VCS must have same configuration across all hosts. The storage for the VM guests must be accessible to all the hosts in the cluster.

See "Network configuration for VCS cluster across physical machines (PM-PM)" on page 58.

See "Sample configuration in a KVM environment" on page 203.

Validating the virtualization environment for virtual machine availability

The VCS utility `havirtverify` validates the virtualization environment. If the virtualization environment is not valid for VCS to manage virtual machines, it logs an error message indicating that the virtualization environment is invalid and resource state is UNKNOWN. Upon receiving this error message, you must correct the virtualization environment and run the `havirtverify` utility manually to validate the environment. Upon successful validation, a verification message displays and the VCS resource state clears in the next monitor cycle.

You can also run this utility manually for verifying the environment.

- Run the `havirtverity` utility manually:

```
# /opt/VRTSvcs/bin/KVMGuest/havirtverify resource_name
```

If validation passes, the following message displays:

```
#/opt/VRTSvcs/bin/KVMGuest/havirtverify resource_name
Red Hat Enterprise Virtualization Environment validation successfully
completed for resource resource_name
```

If validation fails, the following message displays:

```
# /opt/VRTSvcs/bin/KVMGuest/havirtverify resource_name
Virtualization environment validation failed for resource resource_name
```

All the log messages of this utility are sent to the engine log file.

See “Sample configuration in a KVM environment” on page 203.

See “Sample configurations for a Red Hat Enterprise Virtualization (RHEV) environment” on page 206.

Chapter 13

Virtual machine availability for live migration

This chapter includes the following topics:

About live migration

You can enable live migration of guest virtual machines using shared storage or commodity hardware by leveraging Flexible Storage Sharing (FSS) through Cluster Volume Manger (CVM) and Cluster File System (CFS), components of Storage Foundation Cluster File System HA (SFCFSHA). Using CVM significantly reduces planned downtime for individual virtual machines. Individual virtual machines can now be statefully migrated from host to host, enabling better load-balancing, lower machine downtime and path-management of individual physical servers. Physical servers (hosts) can now join and exit the server pool (physical server cluster) at will while the individual guest virtual machines and their corresponding applications continue to run.

For live migration, by using Fast Failover using CVM/CFS in the guest and host, rather than running a single-node Veritas Volume Manager (VxVM) in the host, you

can run the CVM/CFS in the host and cluster multiple physical servers within the same server cluster or server pool. This configuration includes Cluster Server (VCS) also within the host. The significant advantage of creating a cluster of physical servers is that live migration of KVM or RHEV guest virtual machines from one physical server to another is fully operational and supported.

Figure 13-1 Live migration setup for Kernel-based Virtual Machine (KVM)

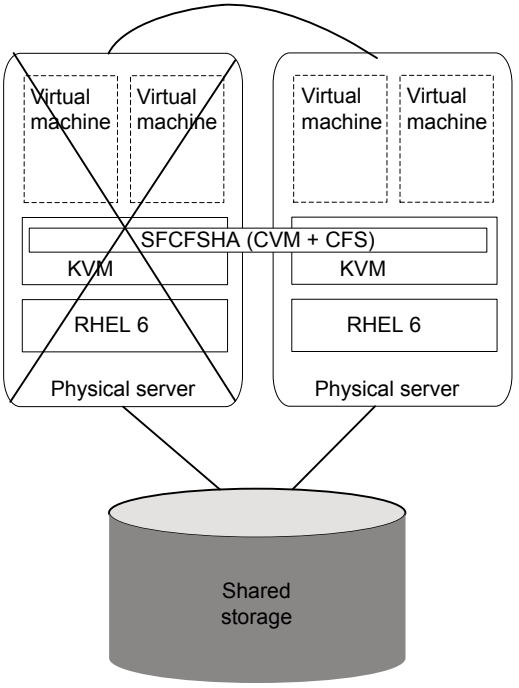
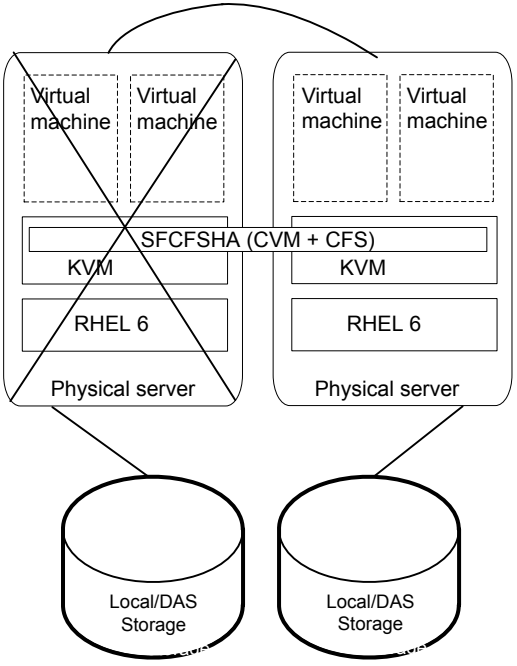


Figure 13-2 Live migration setup for RHEV-based Virtual Machine (RHEV) in FSS configuration



The live migration use case is supported for the following Linux virtualization techologies:

- Red Hat Enterprise Linux (RHEL) KVM
- Red Hat Enterprise Virtualization (RHEV)
- SUSE Linux Enterprise Server (SLES) KVM

Live migration requirements

The following conditions are required for migrating a VM guest from source host to destination host:

- The required guest image must be available on the destination host at the same location.

- The storage and network devices configured in the migrating guest must be identical on source and destination hosts. Any difference may cause the migration process to terminate.
- The KVM hypervisor version on both the hosts should be the same as well as the operating system level.
- For KVM and RHEV environments, you must set password-less SSH communication between the source and destination host.

For detailed information about the required and limitation of virtual machine migration, see your Linux virtualization documentation.

Reduce SAN investment with Flexible Shared Storage in the RHEV environment

Veritas InfoScale Solutions offer the Flexible Storage Sharing (FSS) technology that enables inclusion of SSDs or HDDs to work alongside SAN or DAS in your network. The flexibility to use low-cost SSDs, HDDs alongside SAN network, gives you the opportunity to lower the total cost of ownership (TCO) and provides flexibility for future server or storage investments.

FSS enables network sharing of local storage, DAS or internal, across a global namespace to provide data redundancy, high availability, and disaster recovery without the need of shared storage. Using network interconnect between the nodes, FSS allows network shared storage to co-exist with physically shared storage.

The network sharing of local storage made available by FSS means that physically shared disks are not needed in your storage environment. You can manage your storage environment by cost effectively adding SSDs or HDDS or arrays to your existing storage environment based on storage needs. The total cost of ownership (TCO) for your storage hardware infrastructure is vastly reduced.

FSS has the potential to transform your storage environment without external shared storage or a SAN network.

For more information on administering FSS, refer to the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

Consider the use cases of live migration or disaster recovery of virtual machines in an RHEV environment with FSS enabled for the underlying storage.

For live migration, the virtual machines can use SF components as backend storage configured for FSS. The investments on storage are vastly reduced as FSS lets you use commodity hardware alongside your existing network, serving compute and storage needs from the same servers..

For disaster recovery, VVR provides data replication across dispersed data centres which use Storage Foundation as the backend storage. If the volumes used for replication are created on SF components and the underlying storage is configured for FSS, you get a highly reliable storage management solution that is running on low-cost commodity hardware.

About Flexible Storage Sharing

Flexible Storage Sharing (FSS) enables network sharing of local storage, cluster wide. The local storage can be in the form of Direct Attached Storage (DAS) or internal disk drives. Network shared storage is enabled by using a network interconnect between the nodes of a cluster.

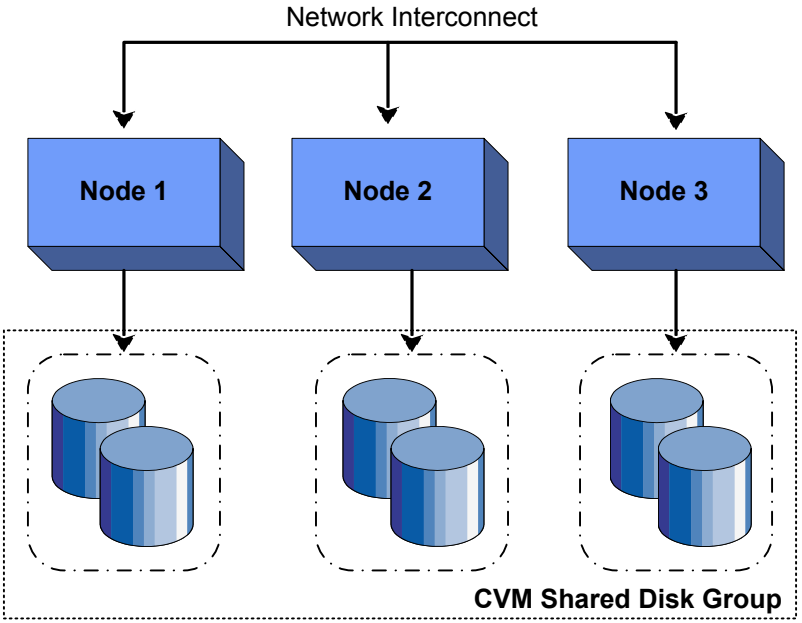
FSS allows network shared storage to co-exist with physically shared storage, and logical volumes can be created using both types of storage creating a common storage namespace. Logical volumes using network shared storage provide data redundancy, high availability, and disaster recovery capabilities, without requiring physically shared storage, transparently to file systems and applications.

FSS can be used with SmartIO technology for remote caching to service nodes that may not have local SSDs.

FSS is supported on clusters containing up to 64 nodes with CVM protocol versions 140 and above. For more details, refer to the *Veritas InfoScale Release Notes*.

Figure 13-3 shows a Flexible Storage Sharing environment.

Figure 13-3 Flexible Storage Sharing Environment



Flexible Storage Sharing use cases

The following list includes several use cases for which you would want to use the FSS feature:

Use of local storage in current use cases	The FSS feature supports all current use cases of the Storage Foundation and High Availability Solutions (Storage Foundation and High Availability Solutions) stack without requiring SAN-based storage.

Off-host processing	Data Migration: ■ From shared (SAN) storage to network shared storage ■ From network shared storage to SAN storage ■ From storage connected to one node (DAS)/cluster to the storage connected to a different node (DAS)/cluster, that do not share the storage Back-up/Snapshots: An additional node can take a back-up by joining the cluster and reading from volumes/snapshots that are hosted on the DAS/shared storage, which is connected to one or more nodes of the cluster, but not the host taking the back-up.
DAS SSD benefits leveraged with existing Storage Foundation and High Availability Solutions features	■ Mirroring across DAS SSDs connected to individual nodes of the cluster. DAS SSDs provides better performance than SAN storage (including SSDs). FSS provides a way to share these SSDs across cluster. ■ Keeping one mirror on the SSD and another on the SAN storage provides faster read access due to the SSDs, and also provide high availability of data due to the SAN storage. ■ There are several best practices for using SSDs with Storage Foundation. All the use-cases are possible with SAN attached SSDs in clustered environment. With FSS, DAS SSDs can also be used for similar purposes.
FSS with SmartIO for file system caching	If the nodes in the cluster have internal SSDs as well as HDDs, the HDDs can be shared over the network using FSS. You can use SmartIO to set up a read/write-back cache using the SSDs. The read cache can service volumes created using the network-shared HDDs.

FSS with SmartIO for remote caching	FSS works with SmartIO to provide caching services for nodes that do not have local SSD devices. In this scenario, Flexible Storage Sharing (FSS) exports SSDs from nodes that have a local SSD. FSS then creates a pool of the exported SSDs in the cluster. From this shared pool, a cache area is created for each node in the cluster. Each cache area is accessible only to that particular node for which it is created. The cache area can be of type, VxVM or VxFS. The cluster must be a CVM cluster. The volume layout of the cache area on remote SSDs follows the simple stripe layout, not the default FSS allocation policy of mirroring across host. If the caching operation degrades performance on a particular volume, then caching is disabled for that particular volume. The volumes that are used to create cache areas must be created on disk groups with disk group version 200 or later. However, data volumes that are created on disk groups with disk group version 190 or later can access the cache area created on FSS exported devices. **Note:** CFS write-back caching is not supported for cache areas created on remote SSDs. For more information, see the document *Veritas InfoScale SmartIO for Solid State Drives Solutions Guide*.
Campus cluster configuration	Campus clusters can be set up without the need for Fibre Channel (FC) SAN connectivity between sites.
FSS in cloud environments	The Flexible Shared Storage (FSS) Technology allows you to overcome the limitations of 'Share-Nothing' storage in cloud environments. FSS enables you to create shared-nothing clusters by sharing cloud block storage over the network.

Limitations of Flexible Storage Sharing

Note the following limitations for using Flexible Storage Sharing (FSS):

- FSS is only supported on clusters of up to 64 nodes.
- Disk initialization operations should be performed only on nodes with local connectivity to the disk.
- FSS does not support the use of boot disks, opaque disks, and non-VxVM disks for network sharing.
- Hot-relocation is disabled on FSS disk groups.
- The VxVM cloned disks operations are not supported with FSS disk groups.
- FSS does not support non-SCSI3 disks connected to multiple hosts.
- Dynamic LUN Expansion (DLE) is not supported.
- FSS only supports instant data change object (DCO), created using the `vxsnap` operation or by specifying "logtype=dco dcoversion=20" attributes during volume creation.
- By default creating a mirror between SSD and HDD is not supported through `vxassist`, as the underlying mediatypes are different. To workaround this issue, you can create a volume with one mediatype, for instance the HDD, which is the default mediatype, and then later add a mirror on the SSD.
  For example:

```
# vxassist -g diskgroup make volume size init=none

# vxassist -g diskgroup mirror volume mediatype:ssd

# vxvol -g diskgroup init active volume
```

Configure Storage Foundation components as backend storage for virtual machines

Veritas supports extension of Storage Foundation (SF) components from the host to guest virtual machines as generic SCSI-3 disks, where SF components are the backing storage for virtual machines. After installing the SFCFSHA cluster, you can configure SF components on individual virtual machines through RHEV-M, which is enabled by the VRTSrhevm CLI package. The guest virtual machines use the exported SCSI-3 disks as backend storage and these must be visible across the cluster.

Live migration of virtual machines is required for cases of workload management, host failure, or a network issue. You can configure VCS on the host to ensure a coordinated live migration in the event of a disruption or maintenance. Without VCS

configured on the host, you need to manually run the `VRTSrhevm` CLI to perform live migration of the virtual machine.

Implementing live migration for virtual machine availability

A virtual machine (VM) can be migrated from one host to another host. This migration can be a live migration or pause migration. You can initiate the migration using:

- The `virsh migrate` command or `virt-manager` console in case of Kernel-based Virtual Machine (KVM) environment
- RHEV-M web interface in case of Red Hat Enterprise Virtualization (RHEV) environment
- The Cluster Server (VCS) `hagrp -migrate` operation (the `hagrp -migrate` command initiates live migration only)

If virtual machine migration is initiated outside VCS (either using the `virsh` commands or the RHEV-M web interface), VCS monitors the migrated guest and can detect the migration process. VCS changes the resource state according to the virtual machine state, i.e. if the guest is live-migrated from one host to another host, the associated KVMGuest resource is brought online on the host where the guest is migrated and on the source node the KVMGuest resource state is reported as OFFLINE (Intentional OFFLINE).

For the KVM environment, Veritas recommends the use of CVM and CFS for live migration where a virtual image needs to be simultaneously accessible on a source and destination node.

See “Sample configuration in a KVM environment” on page 203.

Cluster Server (VCS) has introduced a new migrate operation for initiating service group migration. The KVMGuest agent has implemented a “migrate” entry point to initiate virtual machine migration in KVM and RHEV environment. You can initiate a virtual machine live migration using the `hagrp -migrate` command.

The syntax for the command is:

```
#hagrp -migrate service_group_name -to destination_node_name
```

To verify the password-less SSH requirement for live migration

- Validate password-less SSH by executing following command on source system:

  ```
  # virsh "connect qemu+ssh://destination_node/system; list"
  ```

  If this command asks for a password, then password-less SSH is not set between source and destination node.

  If proper output is returned, then password-less SSH is set properly.

To configure VCS to initiate virtual machine migration

1. To prepare for initiating a virtual machine live migration using `hagrp -migrate` command, you must configure the `PhysicalServer` attribute (system level) of VCS using following command:

   ```
   # hasys -modify  sys_name PhysicalServer physical_server_name
   ```

   For example:

   ```
   # haconf -makerw
   # hasys -modify sys_name PhysicalServer "'hostname'"
   ```

   The `PhysicalServer` name is used while initiating the migration.

2. If `PhysicalServer` attribute is not configured, then the target node name passed to the migrate entry point is used for initiating the migration.

   The KVMGuest Agent `migrate` entry point:

   - For the KVM environment: Agent uses the `virsh migrate` command to initiate virtual machine migration.
   - For the RHEV environment: Agent uses REST APIs to initiate virtual machine migration. It also checks whether the virtual machine migration is allowed or not.

See "About the KVMGuest agent" on page 79.

Note: When a virtual machine is configured for disaster recovery, the virtual machine cannot be migrated across sites.

See "Sample configurations for a Red Hat Enterprise Virtualization (RHEV) environment" on page 206.

Chapter 14

Virtual to virtual clustering in a Red Hat Enterprise Virtualization environment

This chapter includes the following topics:

Installing and configuring Cluster Server for Red Hat Enterprise Virtualization (RHEV) virtual-to-virtual clustering

Red Hat Enterprise Virtualization (RHEV) is a server virtualization solution that uses a KVM hypervisor. As KVM forms a core part of the Linux kernel, this virtualization is highly efficient in Linux environments. Platform management infrastructure and application-specific agents, and other tools are the other components of a RHEV setup.

To enable VCS support for in-guest clustering, before you install VCS on the guest virtual machines, you must set up a private network between them. This involves the following steps:

- Add the two NICs to the virtual machine for private communication

Note: Veritas recommends that you add one more interface/NIC to the virtual machine for public communication. Also, if the virtual machines for which you are configuring the network run on separate physical hosts, ensure that you set up an LLT communication channel between the physical hosts.

- Attach a switch to each of the two additional NICs

To create a network on the physical host

1. From RHEV Manager, create two new logical networks for private LLT heartbeat communication.
2. Assign appropriate physical interfaces to the newly-created logical networks.

To configure a logical network for virtual machines

1. Create two network interfaces, of IntelPro 'e1000' type, and associate them with the newly-created logical networks.
2. Repeat step 1 for each virtual machine where you want to monitor application availability with VCS.

To set up a cluster of virtual (guest) machines with Cluster Server (VCS), perform the following procedures:

- Consult the requirements in:
  Veritas InfoScale Release Notes
- Install InfoScale Availability product on the guest virtual machine. VCS is bundled with the InfoScale Availability product:
  Veritas InfoScale Installation Guide
- Configure VCS in the guest virtual machine
  Cluster Server Configuration and Upgrade Guide

Note: The installation and configuration of VCS inside a virtual machine is similar to that of the physical system. No additional VCS configuration is required to make it work inside the virtual machine.

For more details, see the *Cluster Server Administrator's Guide*.

Fencing support for VCS in-guest clusters

VCS supports SCSI3, non-SCSI3, CP server-based fencing in virtual machines to prevent corruption of data disks.

For information on configuring fencing, see the *Cluster Server Configuration and Upgrade Guide*.

Live migration support

VCS in-guest clustering continues to provide high availability of applications on virtual machines, in live migration scenarios initiated by the virtualization technology.

Veritas has tested for live migration support in the RHEV environment under the following conditions:

- Virtual machine image resides on NFS, iSCSI, or FC storage domain

Storage configuration for VCS in a RHEV environment

To fail over an application from one virtual machine to another, it is mandatory to store the application data on storage shared between the two virtual machines. In an RHEV environment, Veritas has tested application failovers with the application data residing on:

- Dynamic Multipathing (DMP) as a node to virtual machines
- Cluster Volume Manager (CVM) as a device to virtual machines
- Cluster File System (CFS) as a device to virtual machines
- iSCSI LUNs directly attached to the virtual machine
- NFS exported directory mounted inside virtual machine
- Fibre Channel-based LUNs

Note: Veritas recommends using a dedicated virtual network for iSCSI storage.

Chapter 15

Virtual to virtual clustering in a Microsoft Hyper-V environment

This chapter includes the following topics:

Installing and configuring Cluster Server with Microsoft Hyper-V virtual-to-virtual clustering

The Microsoft Hyper-V role is a hypervisor-based server virtualization technology for the x86_64 architecture. It provides you with the software infrastructure and management tools that you can use to create and manage a virtualized server computing environment.

To enable VCS support for in-guest clustering, before you install VCS on the guest virtual machines, you must set up a private network between them. This involves the following steps:

- Add two NICs to the virtual machine for private communication

  Note: Veritas recommends that you add one more interface/NIC to the virtual machine for public communication. Also, if the virtual machines for which you are configuring the network run on separate physical hosts, ensure that you set up an LLT communication channel between the physical hosts.

- Attach a switch to each of the two additional NICs

To create a virtual network on the physical host

1. From the Hyper-V manager, create two virtual networks for private LLT heartbeat communication.
2. Assign appropriate physical interfaces to the newly-created virtual networks.

To configure the network for the virtual machines

1. Create two network interfaces of 'Legacy Network Adaptor' type, and associate them with the newly-created virtual networks.
2. Repeat step 1 for each virtual machine where you want to monitor application availability with VCS.

To set up a cluster of virtual (guest) machines with Cluster Server (VCS), perform the following procedures:

- Consult the requirements in:
  Veritas InfoScale Release Notes
- Install VCS on the guest virtual machine:
  Veritas InfoScale Installation Guide
- Configure VCS in the guest virtual machine
  Cluster Server Configuration and Upgrade Guide

Note: The installation and configuration of VCS inside a virtual machine is similar to that of the physical system. No additional VCS configuration is required to make it work inside the virtual machine.

For more details, see the *Cluster Server Administrator's Guide*.

Fencing support for VCS in-guest clusters

VCS supports non-SCSI3, CP server-based fencing in virtual machines to prevent corruption of data disks. SCSI3 fencing is not supported.

For information on configuring fencing, see the *Veritas InfoScale Installation Guide*.

Live migration support

VCS in-guest clustering continues to provide high availability of applications on virtual machines, in live migration scenarios initiated by the virtualization technology.

Veritas has tested for live migration support in the Hyper-V environment under the following conditions:

- Microsoft Failover Clustering is enabled
- Virtual machine image resides on Microsoft Clustered Shared Volumes

Chapter 16

Virtual to virtual clustering in a Oracle Virtual Machine (OVM) environment

This chapter includes the following topics:

Installing and configuring Cluster Server for Oracle Virtual Machine (OVM) virtual-to-virtual clustering

Oracle VM is an enterprise-grade server virtualization solution that supports guest (virtual machines) that supports various operating systems, including Linux. Based on the Xen hypervisor technology, OVM also provides you with an integrated, Web-based management console.

Before you install VCS on the guest virtual machines, you must set up a private network between them. This involves the following steps:

To enable VCS support of virtual-to-virtual clustering

- Set up a private network between the guest virtual machines.
  - Apart from the public NIC on each physical host, create two additional NICs.

Note: Veritas recommends that you add one more interface/NIC to the virtual machine for public communication. Also, if the virtual machines for which you are configuring the network run on separate physical hosts, ensure that you set up an LLT communication channel between the physical hosts.

If the virtual machines for which you configure the network run on separate physical hosts, ensure that you create a LLT communication channel between the physical hosts.

- Attach a switch to each of the two additional NICs

To create a private network on the physical host

1. From the Oracle VM Manager, create two virtual networks for private LLT heartbeat communication.
2. Assign appropriate physical interfaces to the newly-created virtual networks.

To configure the network for virtual machines

1. Create two interfaces (in a network that is created with the option **Create a hybrid network with bonds/ports and VLANS**) and associate the interfaces with the newly-created virtual networks.
2. Repeat step 1 for each virtual machine where you want to monitor availability with VCS.

To set up a cluster of virtual (guest) machines with Cluster Server (VCS), perform the following procedures:

- Consult the requirements in:
  Veritas InfoScale Release Notes
- Install InfoScale Availability product on the guest virtual machine:
  Veritas InfoScale Installation Guide
- Configure VCS in the guest virtual machine
  Cluster Server Configuration and Upgrade Guide

Note: The installation and configuration of VCS inside a virtual machine is similar to that of the physical system. No additional VCS configuration is required to make it work inside the virtual machine.

For more details, see the *Cluster Server Administrator's Guide*.

Live migration support

Veritas has supported live migration in the OVM environment under the following conditions:

- Virtual machine image resides on NFS data domains

Fencing support for VCS in-guest clusters

VCS supports non-SCSI3, CP server-based fencing in virtual machines to prevent corruption of data disks.

For information on configuring fencing, see the *Cluster Server Configuration and Upgrade Guide*.

Storage configuration for VCS support in Oracle Virtual Machine (OVM)

To fail over an application from one virtual machine to another, it is mandatory to store the application data on storage shared between the two virtual machines. In an OVM environment, Veritas has tested application failovers with the application data residing on:

- Local disks
- Shared Network Attached Storage (NFS)
- Shared iSCSI SANs: abstracted LUNs or raw disks accessible over existing network infrastructure
- Fibre Channel SANs connected to one or more host bus adapters (HBAs)

Note: For more information, see *Oracle* documentation.

VCS in-guest clustering continues to provide high availability of applications on virtual machines, in live migration scenarios initiated by the virtualization technology.

Chapter 17

Disaster recovery for virtual machines in the Red Hat Enterprise Virtualization environment

This chapter includes the following topics:

About disaster recovery for Red Hat Enterprise Virtualization virtual machines

Red Hat Enterprise Virtualization (RHEV) virtual machines can be configured for disaster recovery (DR) by replicating their boot disks using replication methods such as Volume Replicator (VVR), File Replicator (VFR), Hitachi TrueCopy or EMC SRDF. The network configuration for the virtual machines in the primary site may

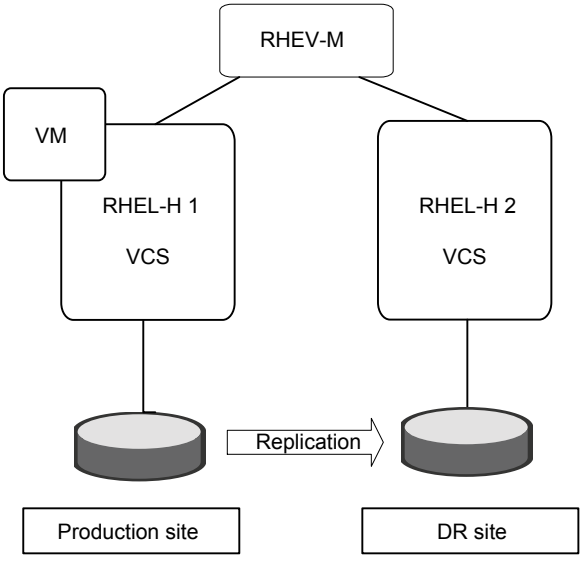
not be effective in the secondary site if the two sites are in different IP subnets. Hence you must make some additional configuration changes to the KVMGuest resource managing the virtual machine.

Supported technologies for replicating virtual machines include:

- Volume Replicator (VVR)
- File Replicator (VFR)
- EMC SRDF
- Hitachi TrueCopy

Note: Live migration of virtual machines across replicated sites is not supported.

Figure 17-1 Schematic of the RHEV DR setup

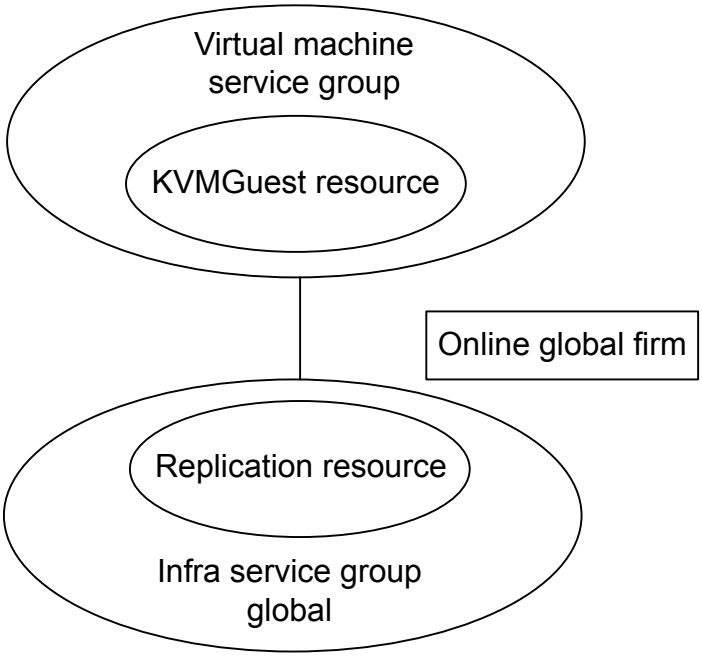


Disaster recovery use cases for virtual machines work in the following way:

- The replication agent takes care of the replication direction. After a disaster event at the primary site, VCS tries to online the replication service group at the secondary site (according to the ClusterFailoverPolicy). The replication resource reverses the replication direction. Reversing the replication direction makes sure that the old secondary LUNs become the new primary LUNs and also are Read-Write enabled on the RHEL-H hosts at the secondary site. This helps RHEV-M activate the Fibre Channel (FC) Storage Domain on the secondary site RHEL-H hosts.

- Before the virtual machine (VM) service group can be brought online, the Storage Pool Manager (SPM) in the datacenter needs to failover to the secondary site. This is achieved by the pre-online trigger script configured on the VM service group. This trigger script checks whether the SPM is still active in the primary RHEV cluster. If so, it deactivates all the RHEL-H hosts in the primary RHEV cluster. Additionally, if the SPM host in the primary RHEV cluster is in the NON_RESPONSIVE state, the trigger fences out the host to enable SPM failover. The trigger script then waits for the SPM to failover to the secondary RHEV cluster. When the SPM successfully fails over to the secondary RHEV cluster, the pre-online trigger script reactivates all the RHEL-H hosts in the primary RHEV cluster, which were deactivated earlier and proceeds to online the VM service group in the secondary site

Figure 17-2 VCS Resource dependency diagram



DR requirements in an RHEV environment

- Licenses for High Availability and Disaster Recovery to run VVR agent.
- Primary and DR site configured with VCS Global Cluster Option.
- VVR and VFR configured between primary and DR site.

- VCS installed on both primary and DR site.

Disaster recovery of volumes and file systems using Volume Replicator (VVR) and Veritas File Replicator (VFR)

In a disaster recovery scenario, you can achieve volume and file level replication by configuring VVR and VFR respectively. Storage Foundation (SF) configured on the hosts provides storage to the guest virtual machines.

VVR and VFR replication technologies replicate volume block devices and file system respectively on the DR site with applications in active state. When the primary site goes down or a network disruption occurs, the VCS Global Cluster Option (GCO) configured for the primary and DR site provides coordinated failover of applications. The DR site takes over the VVR and VFR primary role.

In case you want to move back to the original primary for VVR replication, perform a role transfer. VCS GCO provides the option to select the primary VVR site after a network partition while the applications remain active.

For more information on setting up VVR and VFR, refer to the *Veritas InfoScale™ Solutions Replication Administrator's Guide*.

Why select VVR over array-based replication solutions

Advantages of VVR over array-based replication solutions:

- VVR and VFR replication technologies provide more value and a cost effective solution to alternative costlier array replication technologies in the market.
- VVR can be used on different disk vendor solutions on the primary and the secondary site. For example, VVR works with EMC disks on the primary site and Hitachi disks on the secondary site. VVR does not need the underlying disk configuration to be the same, it only requires the disk space to be the same.
- VxVM, which is a layer below VVR, provides snapshot capabilities and integration with hosts. The snapshot and the host integration capabilities are not available with vendor array-based replication products.
- In comparison to vendor array-based replication solutions, VVR scores more on cost, complexity of management, and high availability. For synchronous replication, you need to evaluate the network costs and complexity.

Consider the use case of disaster recovery of virtual machines across geographically separated data centers. The investments on storage are vastly reduced as FSS allows you to use commodity hardware alongside your existing network. The virtual

machines use Storage Foundation as the backend storage and VVR replicating data written to volumes to the DR site and VFR replicating file system data to the DR site . Overall, you get a highly reliable storage management and replication solution that is running on low cost commodity hardware.

Configure Storage Foundation components as backend storage

Veritas Technologies LLC supports extension of Storage Foundation (SF) components from the host to guest virtual machines as generic SCSI-3 disks, where SF components are the backing storage for virtual machines. After installing the SFCFSHA cluster, you can configure SF components on individual virtual machines through RHEV-M, which is enabled by the VRTSrhevm CLI package. The guest virtual machines use the exported SCSI-3 disks as backend storage and these must be visible across the cluster.

Configure VVR and VFR in VCS GCO option for replication between DR sites

Veritas Volume Replicator (VVR) replicates data written to shared volumes from the primary site to the disaster recovery site. Likewise, Veritas File Replicator (VFR) replicates shared file systems. Configure VCS Global Cluster Option (GCO) to enable failover from primary to the disaster recovery (DR) site. When the primary site goes down or the network link fails, VCS Global Cluster Option (GCO) coordinates the failover of applications to the disaster recovery site.

The VVR agent automatically fails over the applications writing to the volume block devices. However, for file systems, you need to manually fail over the application writing to the file system to the DR site.

For more information on configuring VVR, VFR and VCS GCO, refer to the *Veritas InfoScale™ Solutions Replication Administrator's Guide*.

See "Disaster recovery of volumes and file systems using Volume Replicator (VVR) and Veritas File Replicator (VFR)" on page 159.

Configuring Red Hat Enterprise Virtualization (RHEV) virtual machines for disaster recovery using Cluster Server (VCS)

You can configure new or existing RHEV-based virtual machines for disaster recovery (DR) by setting them up and configuring VCS for DR.

To set up RHEV-based virtual machines for DR

1. Configure VCS on both sites in the RHEL-H hosts, with the GCO option.

   For more information about configuring a global cluster: see the *Veritas InfoScale ™ Solutions Disaster Recovery Implementation Guide*.

2. Configure replication setup using a replication technology such as VVR, VFR, Hitachi TrueCopy, or EMC SRDF.

3. Map the primary LUNs to all the RHEL-H hosts in the primary site.

4. Issue OS level SCSI rescan commands and verify that the LUNs are visible in the output of the `multipath -l` command.

5. Map the secondary LUNs to all the RHEL hosts in the secondary site and verify that they are visible in the output of the `multipath -l` command on all the hosts in the secondary site.

6. Add the RHEL-H hosts to the RHEV-M console.

   - Create two RHEV clusters in the same datacenter, representing the two sites.
   - Add all the RHEL-H hosts from the primary site to one of the RHEV clusters.
   - Similarly, add all the RHEL-H hosts from the secondary site to the second RHEV cluster.

7. Log in to the RHEV-M console and create a Fibre Channel-type Storage Domain on one of the primary site hosts using the primary LUNs.

8. In the RHEV-M console, create a virtual machine and assign a virtual disk carved out of the Fibre Channel Storage Domain created in 7.

   - Configure any additional parameters such as NICs and virtual disk for the virtual machine.
   - Verify that the virtual machine turns on correctly.
   - Install appropriate RHEL operating system inside the guest.
   - Configure the network interface with appropriate parameters such as IP address, Netmask, and gateway.

- Make sure that the NIC is not under network manager control. You can disable this setting by editing the `/etc/sysconfig/network-scripts/ifcfg-eth0` file inside the virtual machine and setting NM_CONTROLLED to "no".
- Make sure that the virtual machine does not have a CDROM attached to it. This is necessary since VCS sends the DR payload in the form of a CDROM to the virtual machine.

9 Copy the package VRTSvcsnr from the VCS installation media to the guest and install it. This package installs a lightweight service which starts when the guest boots. The service reconfigures the IP address and Gateway of the guest as specified in the KVMGuest resource.

To configure VCS for managing RHEV-based virtual machines for DR

1 Install VCS in the RHEL-H hosts at both the primary and the secondary sites.

- Configure all the VCS nodes in the primary site in a single primary VCS cluster.
- Configure all the VCS nodes in the secondary site in the same secondary VCS cluster.
- Make sure that the RHEV cluster at each site corresponds to the VCS cluster at that site.

See Figure 17-2 on page 158.

2 Create a service group in the primary VCS cluster and add a KVMGuest resource for managing the virtual machine. Repeat this step in the secondary VCS cluster.

3 Configure site-specific parameters for the KVMGuest resource in each VCS cluster.

- The `DROpts` attribute enables you to specify site-specific networking parameters for the virtual machine such as IP Address, Netmask, Gateway, DNSServers, DNSSearchPath and Device. The Device is set to the name of the NIC as seen by the guest, for example eth0.
- Verify that the ConfigureNetwork key in the `DROpts` attribute is set to 1.
- The `DROpts` attribute must be set on the KVMGuest resource in both the clusters.

4 Configure the preonline trigger on the virtual machine service group. The preonline trigger script is located at `/opt/VRTSvcs/bin/sample_triggers/VRTSvcs/preonline_rhev`.

- Create a folder in the `/opt/VRTSvcs` directory on each RHEL-H host to host the trigger script. Copy the trigger script in this folder with the name "preonline". Enable the preonline trigger on the virtual machine service group by setting the PreOnline service group attribute. Also, specify the path (relative to `/opt/VRTSvcs`) in the TriggerPath attribute.

For example:

```
group RHEV_VM_SG1 (
    SystemList = { vcslx317 = 0, vcslx373 = 1 }
    ClusterList = { test_rhevdr_pri = 0, test_rhevdr_sec = 1 }
    AutoStartList = { vcslx317 }
    TriggerPath = "bin/triggers/RHEVDR"
    PreOnline = 1
    )
```

For more information on setting triggers, see the *Cluster Server Administrator's Guide*.

5 Create a separate service group for managing the replication direction. This task must be performed for each cluster.

- Add the appropriate replication resource (such as Hitachi TrueCopy or EMC SRDF). For details on the appropriate replication agent, see the *Replication Agent Installation and Configuration Guide* for that agent.
- Add an Online Global Firm dependency from the virtual machine (VM) service group to the replication service group.
- Configure the replication service group as global.

6 Configure the postonline trigger on the replication service group. The postonline trigger script is located at `/opt/VRTSvcs/bin/sample_triggers/VRTSvcs/postonline_rhev`.

- Copy the postonline trigger to the same location as the preonline trigger script, with the name "postonline". Enable the postonline trigger on the replication service group by adding the POSTONLINE key to the TriggersEnabled attribute. Also, specify the path (relative to `/opt/VRTSvcs`) in the TriggerPath attribute.
  For example:

```
group SRDF_SG1 (
    SystemList = { vcslx317 = 0, vcslx373 = 1 }
    ClusterList = { test_rhevdr_pri = 0, test_rhevdr_sec = 1 }
    AutoStartList = { vcslx317 }
    TriggerPath = "bin/triggers/RHEVDR"
```

```
    TriggersEnabled = { POSTONLINE }
)
```

For more information on setting triggers, see the *Cluster Server Administrator's Guide.*

If you have multiple replicated Storage Domains, the replication direction for all the domains in a datacenter must be the same.

To align replication for multiple replicated Storage Domains in a datacenter

1. Add all the replication resources in the same Replication Service Group.
2. If you require different Storage Domains to be replicated in different directions at the same time, configure them in a separate datacenter.

   This is because the Storage Pool Manager (SPM) host requires read-write access to all the Storage Domains in a datacenter.

After completing all the above steps, you can easily switch the virtual machine service group from one site to the other. When you online the replication service group in a site, the replication resource makes sure that the replication direction is from that site to the remote site. This ensures that all the replicated devices are read-write enabled in the current site.

See "About disaster recovery for Red Hat Enterprise Virtualization virtual machines" on page 156.

Disaster recovery workflow

1. Online the replication service group in a site followed by the virtual machine service group.
2. Check the failover by logging into the RHEV-M console. Select the **Hosts** tab of the appropriate datacenter to verify that the SPM is marked on one of the hosts in the site in which the replication service group is online.
3. When you bring the Replication Service Group online, the postonline trigger probes the KVMGuest resources in the parent service group. This is to ensure that the virtual machine service group can go online.
4. When you bring the virtual machine service group online, the preonline trigger performs the following tasks:

   - The trigger checks whether the SPM is in the local cluster. If the SPM is in the local cluster, the trigger checks whether the SPM host is in the UP state. If the SPM host is in the NON_RESPONSIVE state, the trigger fences out the host. This enables RHEV-M to select some other host in the current cluster.

- If the SPM is in the remote cluster, the trigger deactivates all the hosts in the remote cluster. Additionally, if the remote SPM host is in the NON_RESPONSIVE state, the trigger script fences out the host. This enables RHEV-M to select some other host in the current cluster.
- The trigger script then waits for 10 minutes for the SPM to failover to the local cluster.
- When the SPM successfully fails over to the local cluster, the script then reactivates all the remote hosts that were previously deactivated.
- Then the trigger script proceeds to online the virtual machine service group.

5 When the KVMGuest resource goes online, the KVMGuest agent sets a virtual machine payload on the virtual machine before starting it. This payload contains the site-specific networking parameters that you set in the DROpts attribute for that resource.

6 When the virtual machine starts, the `vcs-net-reconfig` service is loaded and reads the DR parameters from the CDROM and then applies them to the guest. This way, the networking personality of the virtual machine is modified when the virtual machine crosses site boundaries.

Troubleshooting a disaster recovery configuration

◆ You can troubleshoot your disaster recovery in the following scenarios:

- When the service groups are switched to the secondary site, the hosts in the primary site may go into the NON_OPERATIONAL state. To resolve this issue, deactivate the hosts by putting them in maintenance mode, and reactivate them. If the issue is not resolved, log onto the RHEL-H host and restart the `vdsmd` service using the `service vdsmd restart`command. If the issue still persists, please contact RedHat Technical Support.
- After a DR failover, the DNS configuration of the virtual machine may not change. To resolve this issue, check if the network adapter inside the virtual machine is under Network Manager control. If so, unconfigure the network adapter by editing the `/etc/sysconfig/network-scripts/ifcfg-eth0` file inside the virtual machine and setting NM_CONTROLLED to "no".
- After a failover to the secondary site, the virtual machine service group does not go online. To resolve this issue, check the state of the SPM in the data center. Make sure that the SPM is active on some host in the secondary RHEV cluster. Additionally, check the VCS engine logs for more information.

Chapter 18

Multi-tier business service support

This chapter includes the following topics:

About Virtual Business Services

The Virtual Business Services feature provides visualization, orchestration, and reduced frequency and duration of service disruptions for multi-tier business applications running on heterogeneous operating systems and virtualization technologies. A virtual business service represents the multi-tier application as a consolidated entity that helps you manage operations for a business service. It builds on the high availability and disaster recovery provided for the individual tiers by Veritas InfoScale products such as Cluster Server.

Application components that are managed by Cluster Server or Microsoft Failover Clustering can be actively managed through a virtual business service.

You can use the Veritas InfoScale Operations Manager Management Server console to create, configure, and manage virtual business services.

Sample virtual business service configuration

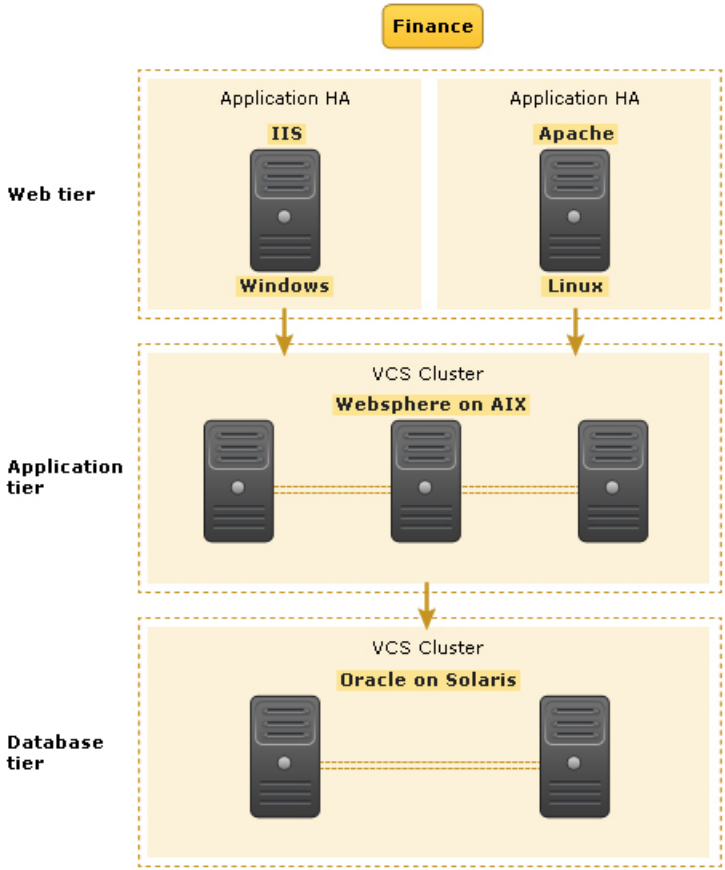
This section provides a sample virtual business service configuration comprising a multi-tier application. Figure 18-1 shows a Finance application that is dependent

on components that run on three different operating systems and on three different clusters.

- Databases such as Oracle running on Solaris operating systems form the database tier.
- Middleware applications such as WebSphere running on AIX operating systems form the middle tier.
- Web applications such as Apache and IIS running on Windows and Linux virtual machines form the Web tier.

  Each tier can have its own high availability mechanism. For example, you can use Cluster Server for the databases and middleware applications for the Web servers.

Figure 18-1 Sample virtual business service configuration



Each time you start the Finance business application, typically you need to bring the components online in the following order – Oracle database, WebSphere, Apache and IIS. In addition, you must bring the virtual machines online before you start the Web tier. To stop the Finance application, you must take the components offline in the reverse order. From the business perspective, the Finance service is unavailable if any of the tiers becomes unavailable.

When you configure the Finance application as a virtual business service, you can specify that the Oracle database must start first, followed by WebSphere and the Web servers. The reverse order automatically applies when you stop the virtual business service. When you start or stop the virtual business service, the components of the service are started or stopped in the defined order.

For more information about Virtual Business Services, refer to the *Virtual Business Service–Availability User's Guide*.

Recovery of Multi-tier Applications managed with Virtual Business Services in Veritas Operations Manager

In a multi-tier business service, different tiers usually have different requirements. One tier may require full-fledged high availability with split-second error detection and fast failover, while other tiers just need basic start and stop capability. The management of start and stop for any service is critical to successful recovery. Business services have strict start and stop orders that need to be validated before proceeding to the next service. Often times, these services are managed by different IT teams. The actual start/stop command for each tier may be simple, but given the amount of coordination, communication, validation and handover between the different teams, the process can be time consuming.

The Red Hat Enterprise Virtualization environment with Cluster Server can be managed with Veritas Operations Manager (VOM), which provides a centralized console to monitor, visualize, and configure all resources. VOM also provides a view into every level of the IT infrastructure from the application to the disk drives. It provides a consistent Graphical User Interface (GUI) and Command Line Interface (CLI) driven administration across all platforms: Red Hat Enterprise Virtualization, Red Hat Enterprise Linux, VMware, UNIX and Windows. VOM reports on the relationship of applications to virtual machines, physical servers and clusters. Many organizations use different staff to manage servers, SAN and storage connectivity, storage and applications. These organizations benefit from this unified view that can administer server clusters and HA/DR configurations from this single console view.

Service Group Management in Virtual Business Services

Service group management improves business resiliency by providing a method to bundle hardware, software, applications, databases and networks into a single entity with dependencies. By monitoring the health and performance of these service groups, through proactive notification, pending issues can be quickly addressed. VOM reports on the relationship of applications to virtual machines, physical servers and clusters and provides coordinated failover of services that span virtual machines and physical machines for multi-tier applications. In the past, customers who wanted this functionality had to build scripts to automate these procedures but this method was complex to manage and test.

To help customers address these issues, Veritas introduced Virtual Business Services (VBS). Virtual Business Services combines the power of VCS, AppHA and VOM to provide complete multi-tier business service management and High Availability. VBS now enables management of multi-tier business services on top of VOM and VCS which allows VOM to be used as a single tool for availability management.

Virtual Business Services achieves the following:

- Co-ordinates the start and stop across different operating systems and/or platforms
- Provides fault management and propagation between tiers
- Manages multi-tier Disaster Recovery support
- Enables automated Disaster Recovery of a complete Virtual Business Service and Virtual Machine management support (start and stop)

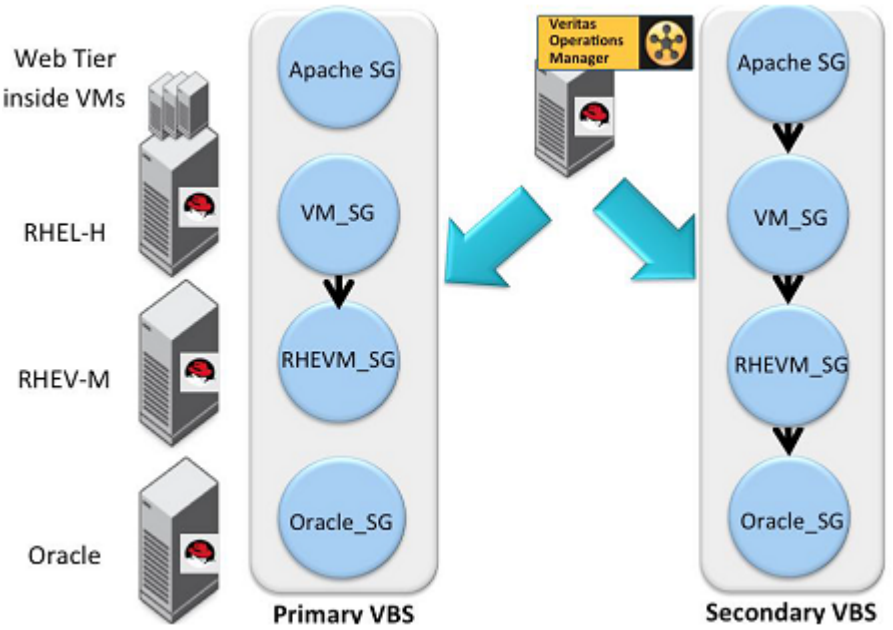
High Availability is primarily managed within each tier. The cluster is responsible to keep services highly available within the cluster. The boundaries for an application are the cluste instance. Logically, a VBS can be seen as a container that allows service groups to be built into a single object. To enable VBS, Red Hat Enterprise Virtualization Environments must have VCS installed on the physical server. For the other operating environments within the VBS, it is required that each tier has VCS, Microsoft Cluster Server installed.

In order to deploy VBS, there must be at least one VOM Central Server installed in the data center. The VOM Central Server is used for configuration, visualization and management of VBS. However, after the initial configuration of a VBS, it can be managed using a CLI as well. VBS functionality does not depend on VOM Central Server. CLI operations work regardless of whether the VOM Central Server is available or not, and the member nodes of a VBS will operate autonomously of the VOM Central Server once VBS is deployed.

Application DR can be between VMs or from a Virtual to Physical DR and vice versa. During the failover of Virtual Machine there is an automatic update of VM (IP, DNS, netmask) to ensure user access to the new instance.

An example of how DR operates across a multi-tier environment

Figure 18-2 DR in a multi-tier environment



Veritas Operations Manager also includes the ability to associate different Virtual Business Services into a Disaster Recovery Plan. This feature enables another level of automation because it allows the customer to combine service groups, Virtual Business Groups and manual scripts into a single procedure. It provides the sequence of operations that will be performed at the DR site, in the event of a disaster. The GUI allows you to choose items to include into the plan and provides single click failover of an entire data center to a secondary site.

Chapter 19

Managing Docker containers with InfoScale Enterprise

This chapter includes the following topics:

About managing Docker containers with InfoScale Enterprise product

You can deploy and manage Docker containers that host applications by leveraging the storage management and high availability features of the InfoScale Enterprise product. The Veritas File System (VxFS) and Veritas Volume Manager (VxVM) components of the InfoScale Enterprise product provide storage management capabilities to manage Docker containers. The Cluster Server Agent for Docker provides high availability to the Docker Daemon and Docker Containers. If a node goes down or during system outage, the agent fails over Docker Daemon and Docker Container to another node in the cluster.

The InfoScale Enterprise product can be used to replicate and migrate Docker Containers. This replication capability is provided by VVR and VFR technologies, which are part of the InfoScale Enterprise product. Docker containers can be recovered across geographically distributed sites by configuring the Global Cluster Option (GCO) in cluster deployments of VCS.

As Docker technology continues to evolve as one of the leading container technology, the InfoScale Enterprise product solves a few key use cases in Docker environments.

The InfoScale Enterprise product addresses the following use cases:

- Scale storage capacity for Docker containers
- Manage data for Docker containers
- Migrate Docker containers
- Provide High Availability for containers
- Provide Disaster Recovery for Docker containers

About the Cluster Server agents for Docker, Docker Daemon, and Docker Container

The Cluster Server (VCS) agents monitor specific resources within an enterprise application. They determine the status of resources and start or stop them according to external events. The Cluster Server agents for Docker Daemon and Docker Container provide high availability for Docker daemon and Docker containers in a cluster.

The Docker daemon provides the base infrastructure for creating and hosting Docker containers. The Docker Daemon can also perform storage-related operations such as creating volumes, mounting or unmounting file systems, removing volumes and other storage related activities. The Docker Daemon eases deployment of Docker Containers without the need to manually provision storage to containers. But, you can also choose to manually provision storage to Docker Containers.

The Docker Container agent monitors the Docker Container instances while they are online and offline. If the system fails, the agent detects failure and takes the container instances offline. Cluster Server initiates failover to another system in the cluster and the agent brings the container instances online.

Supported software

For information on the software versions that the Cluster Server agents for Veritas support, see the Veritas Services and Operations Readiness Tools (SORT) site: http://sort.veritas.com/agents.

How the agents makes Veritas highly available

The Cluster Server agent for Docker Daemon continuously monitors the Docker Daemon process to check the status of the Docker Service. The agent for Docker Daemon is Intelligent Monitoring Framework (IMF) aware and uses Asynchronous Monitoring Framework (AMF) kernel driver for IMF notification.

The Cluster Server agent for Docker Container monitors the configured container using the `docker inspect` command. The agent for Docker Container is IMF aware and uses AMF kernel driver for IMF notification.

Documentation reference

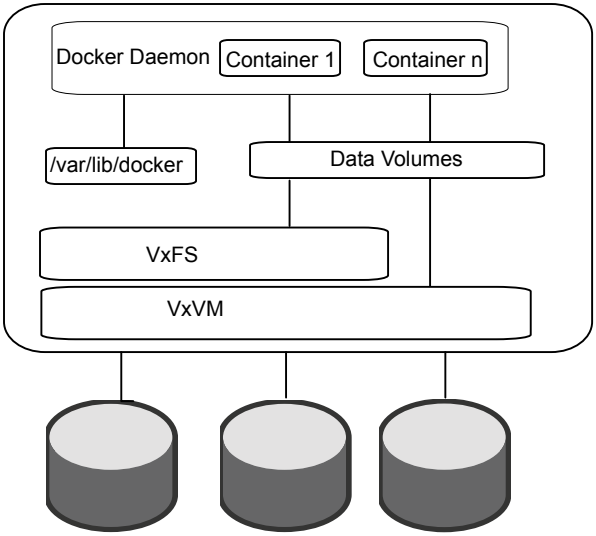
You can access the *Cluster Server Agent for Docker Installation and Configuration Guide*, at https://sort.veritas.com/agents/detail/2634.

Managing storage capacity for Docker containers

Docker containers that host applications need underlying storage to store and manage application data and metadata related to the containers. Storage can either come from disk arrays or Direct Attached Storage (DAS) devices. There are three ways to provision storage for Docker containers.

- Manually provision storage to a Docker daemon which in turn assigns storage to Docker containers.
  See "Provisioning storage for Docker infrastructure from the Veritas File System" on page 175.
- Directly provision Docker volumes to Docker containers without the need of provisioning storage from Docker daemons.
  See "Provisioning data volumes for Docker containers" on page 176.
- Automatically provision storage for Docker containers through the Vertias Volume driver.
  See "Automatically provision storage for Docker Containers" on page 178.

Figure 19-1 Scaling Docker containers on a stand-alone host



Provisioning storage for Docker infrastructure from the Veritas File System

You can provision storage for Docker infrastructure on VxFS. The Docker infrastructure is managed by Docker Daemon. One of the ways to provision storage for Docker containers is from the Docker infrastructure. You can create and run Docker containers by provisioning storage from the Docker infrastructure. The Docker Daemon uses the '/var/lib/docker' directory on the VxFS file system for creating Docker infrastructure.

Alternatively, storage from VxFS and VxVM can be provisioned as data volumes to Docker containers. You can choose not to create the `/var/lib/docker` directory on VxFS, by editing the `/etc/sysconfig/docker` file with an alternate directory to be used by the Docker Daemon.

To provision Docker infrastructure from the default Docker directory

1. Mount the Docker infrastructure directory on VxFS.

```
# vxdg init dockerdg disk1 disk2
# vxassist -g dockerdg make dockervol 200G
# mkfs -t vxfs /dev/vx/dsk/dockerdg/dockervol
# mount -t vxfs /dev/vx/dsk/dockerdg/dockervol /var/lib/docker/
```

2. Start Docker Daemon and create container. Note the change in the size of directory.

```
# systemctl start docker
```

To provision Docker infrastructure from another directory

1. Create a VxFS directory.

```
# vxdg init dockerdg disk1 disk2
# vxassist -g dockerdg make dockervol 1G
# mkfs -t vxfs /dev/vx/dsk/dockerdg/dockervol
# mount -t vxfs /dev/vx/dsk/dockerdg/dockervol /dockervol
```

2. In the '/etc/sysconfig/docker' file, append the `OPTIONS` field with `-g dockervol`.

```
# grep OPTIONS /etc/sysconfig/docker
```

Note: If SELinux is already disabled, do not append the docker file with the option `--selinux-disabled`.

3. Start the Docker Daemon and note the directory size.

```
# systemctl start docker
```

Provisioning data volumes for Docker containers

As seen in the earlier sections, one of the ways to provision storage to Docker containers is through creating a Docker infrastructure on the file system. However, provisioning storage to Docker containers is not limited to provisioning through the Docker infrastructure. You can provision storage from VxFS and VxVM as data volumes to containers.

Data volumes provisioned from VxFS ensures that container data is persistent. Cluster File System enables data sharing among containers across the cluster. Data updates to volumes are made directly on the VxFS mount point or VxVM

volumes. You can use all the VxFS and VxVM features to manage these data volumes.

Provisioning storage on Veritas File System as data volumes for containers

You can provision data volumes to the container using the -v flag in the `docker run` command. To provision multiple data volumes, pass the –v flag multiple times in the `docker run` command.

To export a data volume from VxFS as backend storage

1. On the host node create a VxVM volume and mount it as a VxFS file system.

   `# vxassist -g dockerdg make containervolume 1G`

   `# mkfs -t vxfs /dev/vx/dsk/dockerdg/containervolume`

   `# mkdir /containervolume`

   `# mount -t vxfs /dev/vx/dsk/dockerdg/containervolume /containervolume`

   Where *containervolume* is the storage provisioned to Docker containers.

2. Mount the volume inside the container.

   `# docker run -it --name vm-container -v /containervolume:/vol ubuntu /bin/bash`

3. The default permissions for a volume mounted to a Docker container is read-write. However, you can change the access permissions of volume while mounting it to the container.

   `# docker run -it --name vm-container -v /containervolume:/vol -v /containervolume1:/vol1:ro ubuntu /bin/bash`

Provisioning VxVM volumes as data volumes for containers

Depending on the application you are running, you may choose to export VxVM volumes as data volumes inside a container.

Export a VxVM data volume to a container using the -v flag with the `docker run` commands . You can mount multiple data volumes by passing the –v flag multiple times in the `docker run` command.

In order to use a raw VxVM volume inside a container, either assign privilege permissions to the Docker container or pass the raw volume as a `--device` option in the `docker run` command.

To export a VxVM data volume to containers:

- On the host system create volume and mount it.

  ```
  # vxassist -g dockerdg make containervolume 1G
  ```

  ```
  # docker run -it -v /dev/vx/dsk/dockerdg/containervolume:/mnt/vol
  rhel bash
  ```

  The VxVM volume is mounted inside the container at `/mnt/vol`.

Creating a data volume container

Create a data volume container from a VxFS mount point or a VxVM volume. The data volume container is a directory on the VxFS file system on the host node. You can export volumes from the data volume container to other containers on the node.

To create a data volume container:

1. Create a VxFS directory /dockervol on the host node.

   ```
   # vxdg init dockerdg hitachi_vsp0_03f1 hitachi_vsp0_03f2
   ```

   ```
   # vxassist -g dockerdg make dockervol 1G
   ```

   ```
   # mkfs -t vxfs /dev/vx/dsk/dockerdg/dockervol
   ```

   ```
   # mkdir /dockervol
   ```

   ```
   # mount -t vxfs /dev/vx/dsk/dockerdg/dockervol /dockervol
   ```

2. Create a data volume container.

   ```
   # docker run -it -v /dockervol:/dockervol --name
   datavolumecontainer rhel7 /bin/true
   ```

3. Create another data volume container and use the data volumes from the first data volume container.

   ```
   # docker run -it --volumes-from datavolumecontainer --name
   datavolumecontainer1 rhel7 /bin/bash
   ```

Automatically provision storage for Docker Containers

The Veritas InfoScale volume driver plugin for Docker extends the capability of Docker daemon to handle storage-related operations such as creating volumes or file systems, mounting or unmounting file systems, removing volumes and so on. With this plugin, docker containers can be started with storage attached to them automatically, which helps in ease of deployment of Docker containers. The Veritas driver supports Docker version 1.9 or later. It also integrates with docker volume

CLI. It also seamlessly works with Docker Swarm technology that allows container orchestration.

The prerequisite is to install the Veritas InfoScale Docker volume plugin rpm and configure a disk group before you create volumes.

Download the latest plugin from this location:

https://sort.veritas.com/storage_management

Installing the Veritas InfoScale Docker volume plugin

Install the VRTSdocker-plugin rpm available from the Veritas InfoScale products install bundle on all the nodes on which the docker volume plugin is meant to be installed. This package enables docker containers to use the persistent storage that is provided by Veritas.

- Run the following command.

  # `rpm -ivh VRTSdocker-plugin-<version>-Linux.x86_64.rpm`

  where <*version*> is the version number of the plugin.

Configuring a disk group

Create a disk group for a shared storage environment or a non-shared environment.

1. For shared storage environment, create a shared disk group with name 'dockerdg'.

   # `/usr/sbin/vxdg -s init dockerdg <disk_1> <disk_2> <disk_3> .. <disk_n>`

2. For non-shared storage environment, create a shared disk group with name 'dockerdg'.

   # `/usr/sbin/vxdg -s -o fss init dockerdg <disk_1> <disk_2> <disk_3> .. <disk_n>`

For more information on how to create a VxVM disk group, refer to the `vxdg` manual page.

Creating Docker Containers with storage attached automatically

To create docker containers with storage attached automatically

In the procedure, `demovol` is the name of the created volume.

1. Create a volume using the docker volume create command.

   ```
   # docker volume create -d veritas --name demovol
   ```

2. Alternatively, you can also specify size and or layout options using `-o` option.

   ```
   # docker volume create -d veritas --name demovol -o size=1g -o
   layout=mirror
   ```

   Note: If options are not provided while creating a volume, then it uses default values for size and layout that are stored in the `/etc/vx/docker/vxinfoscale-default.conf` file. If `layout=auto` is mentioned, then InfoScale automatically chooses the best layout for the environment.

3. You can verify whether the volume is created properly by running the a `docker volume ls` command.

   ```
   # docker volume ls


   DRIVER    VOLUME NAME
   veritas   demovol
   ```

4. You can verify the detail information of the volume using `docker volume inspect` command.

   ```
   # docker volume inspect demovol


   [
   {
   "Name": "demovol",
   "Driver": "veritas",
   "Mountpoint": "/dockerfs/demovol_dockerdg",
   "Labels": {},
   "Scope": "global"
   }
   ]
   ```

5 Launch docker container.

```
# docker run --name <container_name> -it --volume-driver veritas
-v demovol:/vol<docker_image> <command>
```

Where, VxVM volume 'demovol' gets automatically attached to the docker container.

6 You may verify that the container is running and accessible.

```
# docker ps <container_name>
```

7 You can also remove the volume if it is not needed anymore.

```
# docker volume rm demovol
```

Avoid noisy neighbor problem by using Quality of Service support

You can set the `maxiops` limit on a volume to avoid noisy neighbor problem.

To set the maximum IOPS for a volume

- Provide the `maxiops` value while creating a volume.

```
# docker volume create -d veritas --name <volume-name> -o
maxiops=<IOPS limit>
```

For example, `# docker volume create -d veritas --name demovol -o maxiops=10000`

Provision to create snapshots

You can create space optimized snapshots of a volume by using the docker volume CLI.

- To create a snapshot of volume vol1 named snapvol1.

```
# docker volume create -d veritas --name snapvol1 -o sourcevol=vol1
[-o cachesize=<cachesize>]
```

Note: The option *cachesize* is optional. If this option is not provided then space optimized snapshot with the default *cachesize* is created. Currently the default *cachesize* value is 30% of the source volume size.

Configuring Veritas volume plugin with Docker 1.12 Swarm mode

Veritas volume plugin seamlessly works with Docker Swarm which allows container orchestration.

The following procedure uses

To configure Veritas volume plugin with Docker 1.12 Swarm mode

1. Consider a docker swarm cluster of two nodes: docker1 and docker2.

```
# docker node ls


ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS
bd3ccjzm4qmo1ntil88r9q0la * docker1 Ready Active Leader
d3rbrj0d4goyfckae0wozwwew docker2 Ready Active
```

2. Create a Veritas volume.

```
# docker volume create -d veritas --name volume1 -o size=500m
```

3. Create a MYSQL service from the swarm manager by providing source volume name.

   Use `volume1` as the source volume name that was created using the Veritas driver.

```
# docker service create --replicas 1 --name sql1 --mount
type=volume,source=volume1,target=/var/lib/mysql,readonly=false
-e MYSQL_ROOT_PASSWORD=root123 mysql

# docker service ps sql1


ID NAME                    IMAGE  NODE
6e2dlvx27iwrgrwdcdf43u4d9 sql1.1 mysql docker1

DESIRED STATE  CURRENT STATE  ERROR
Running        Running        44 seconds ago
```

   Where, mysql service is scheduled on node docker1.

4 Write some persistent data to the mysql database on node docker1.

```
# docker ps -a


CONTAINER ID IMAGE         COMMAND
d844dfa66f65 mysql:latest  "docker-entrypoint.sh"

CREATED        STATUS
A minute ago   Up

PORTS      NAMES
3306/tcp   sql1.1.6e2dlvx27iwrgrwdcdf43u4d9
```

5 [root@docker1] # `docker exec -it d844dfa66f65 bash`

6 root@d844dfa66f65: `/# mysql -proot123`

```
mysql> create database swarm_test;
Query OK, 1 row affected (0.02 sec)

mysql> use swarm_test;
Database changed

mysql> create table people (name text, age integer);
Query OK, 0 rows affected (0.04 sec)

mysql> insert into people values ('Person1', 29);
Query OK, 1 row affected (0.00 sec)

mysql> insert into people values ('Person2', 31);
Query OK, 1 row affected (0.01 sec)

mysql> select * from people;
+---------+------+
| name | age |
+---------+------+
| Person1 | 29 |
| Person2 | 31 |
+----------+------+
2 rows in set (0.00 sec)
```

7 Simulate node failure on docker1. The MySQL service gets re-scheduled on another node by Docker Swarm.

[root@docker1]# `docker node update --availability drain docker1`

8 [root@docker1]# `docker service ps sql1`

```
ID NAME                    IMAGE        NODE
8rofbg2td0i7oubzyxpv0kvik sql1.1 mysql docker2
6e2dlvx27iwrgrwdcdf43u4d9 \_ sql1.1 mysql docker1

DESIRED STATE CURRENT STATE ERROR
Running       Running      47 seconds ago
Shutdown  Shutdown   about a minute ago
```

The MySQL service gets re-scheduled on node docker2 by Docker Swarm.

9 Verify on node docker2 that container with MySQL service is created and verify updated data in the database.

[root@docker2] # `docker ps -a`

```
CONTAINER ID  IMAGE   COMMAND
9fafb70c793b mysql:latest "docker-entrypoint.sh"


CREATED           STATUS
About a minute ago Up

PORTS NAMES
3306/tcp sql1.1.8rofbg2td0i7oubzyxpv0kvik
```

[root@docker2] # `docker exec -it 9fafb70c793b bash`

```
root@9fafb70c793b:/# mysql -proot123



mysql> use swarm_test;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed

mysql> select * from people;
+---------+------+
| name | age |
+---------+------+
| Person1 | 29 |
| Person2 | 31 |
+----------+------+
2 rows in set (0.00 sec)
```

In this procedure, as InfoScale storage made data volume available on the other node, container migration operation is successful.

About using InfoScale Enterprise features to manage storage for containers

You can utilize features of InfoScale Enterprise, such as snapsots, volume layouts, and so on to effectively manage data volumes provisioned for docker containers in order to improve storage efficiency, redundancy of containers.

For IOPs intensive workloads running in Docker environments, there is a need to intelligently adopt to the storage capacity requirement based on application load. The Flexible Storage Sharing (FSS) feature of InfoScale Enterprise enables you to keep adding DAS storage devices to the environment. This flexibility to add DAS devices and export a local storage device over the network, allows the product to adopt to the capacity demands by applications running inside Docker Containers. Similarly, the SmartIO feature provides SSD caching to improve performance of applications running inside Docker Containers.

For more on the features of the InfoScale Enterprise product, refer to the *Storage Foundation Administrator's guide*.

Offline migration of Docker containers

To migrate Docker Containers and or Docker Daemon, you need to configure Docker Daemon and containers for Cluster File System. After migration, CFS ensures that data accessed by containers is consistent across the cluster.

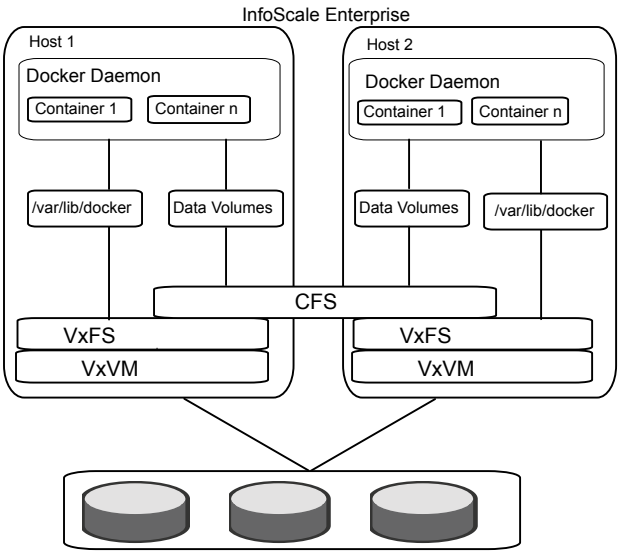
Currently, Docker does not support live migration.

Migrating Docker containers

The InfoScale Enterprise product addresses the use cases where you only want to migrate Docker containers, which means only to migrate data. Since the Docker Daemon is not migrated, the node does not need to run the Cluster Server Agent for Docker Containers.

This section lists the manual steps to migrate Docker containers.

Figure 19-2 Migration of Docker containers



To configure the Docker containers using SFCFS

1 Create a shared disk group and volume and do mkfs.

```
# vxdg -s init dockdg disk1 disk2 disk3
# vxassist -g dockdg make vol7 10G
# mkfs -t vxfs /dev/vx/dsk/dockdg/vol7
```

2 Mount the volume on each cluster node.

```
# mount -t vxfs -o cluster /dev/vx/dsk/dockdg/vol7 /containervolume
```

3 Write to a file on the mount point inside container.

```
# docker run -it -v /containervolume:/datavolume
rhel6 /bin/bash
```

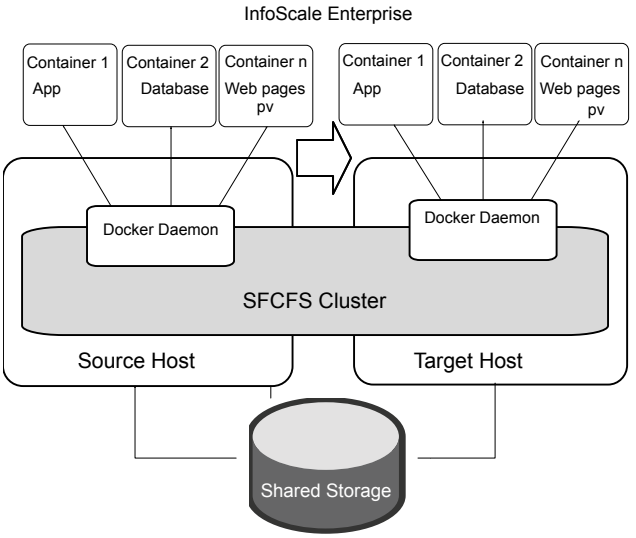
Where */containervolume* is the storage provisioned to Docker containers and it is accessed inside containers under the */datavolume* directory.

Migrating Docker Daemons and Docker Containers

InfoScale Enterprise also addresses the use case related to migrating both the infrastructure and container data. In this scenario, you need to bring down the Docker Daemon and Docker Container on the primary node and bring them up on the secondary node.

While migrating containers, ensure that you stop the Docker Daemon and Docker Container on the primary node before you migrate them to the secondary node.

Figure 19-3 Migration of Docker Daemon and Docker Containers



To configure the Docker Daemon and containers using SFCFS

1 Create a shared disk group and volume and do mkfs.

```
# vxdg -s init dockdg disk1 disk2 disk3
# vxassist -g dockdg make vol7 10G
#  mkfs -t vxfs /dev/vx/dsk/dockdg/vol7
```

2 Mount the volume on each cluster node.

```
# mount -t vxfs -o cluster /dev/vx/dsk/dockdg/vol7 /docvol
```

Where *docvol* is the directory used by the Docker Daemon for Docker infrastructure.

3 Configure Docker to start the Docker Daemon on newly created CFS mount point, `/docvol`.

See "Provisioning storage for Docker infrastructure from the Veritas File System" on page 175.

Note: Docker stores all its data, metadata, containers on alloted CFS shares. Though this CFS share is visible on all cluster nodes, Docker Daemon should be started from only one node a time.

4 To initialize the migration make sure Docker Daemon is stopped from the source node and CFS share is not being consumed by Docker Daemon.

```
# systemctl stop docker
```

5 Start the Docker Daemon on target node by appending `-g /docvol` in the `OPTIONS` field in the `/etc/sysconfig/docker` configuration file..

```
# systemctl start docker
```

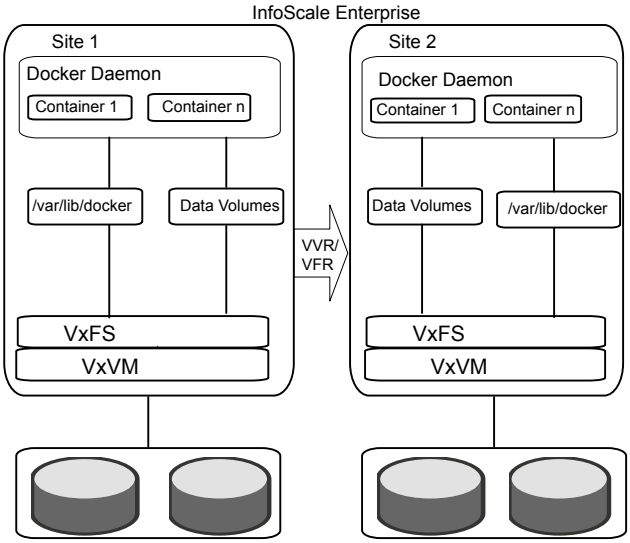
The Docker Daemon is initialized on the target node. After migration is complete, Docker containers will be in exited state. User may need to manually start the containers.

Note: To configure Docker Containers with VCS, refer to the *Cluster Server Agent for Docker Installation and Configuration Guide* guide.

Disaster recovery of volumes and file systems in Docker environments

InfoScale Enterprise helps you replicate both Docker infrastructure managed by Docker Daemon and data volumes of Docker Containers or choose to replicate only the Docker infrastructure or the data volumes. After replicating the data, metadata associated to containers, InfoScale Enterprise migrates the data volumes and infrastructure to the disaster recovery site.

Figure 19-4 Disaster recovery with VVR and VFR technologies



VVR and VFR replication technologies replicate volume block devices and file system respectively on the DR site with applications in active state. When the primary site goes down or a network disruption occurs, the VCS Global Cluster Option (GCO) configured for the primary and DR site provides coordinated failover of applications. The DR site takes over the VVR and VFR primary role.

In case you want to move back to the original primary for VVR replication, perform a role transfer. VCS GCO provides the option to select the primary VVR site after a network partition while the applications remain active.

For more information on setting up VVR and VFR, refer to the *Storage Foundation and High Availability Solutions Replication Administrator's Guide*.

Configuring Docker containers for disaster recovery

Setting up disaster recovery for backing filesystem and volumes of the Docker container and Docker Daemon using VVR/VFR is similar to setting up disaster recovery plans for a physical host or a virtual machine.

To set up replication using VVR

In this section, we are using PrimarySite and SecondarySite to indicate primary and secondary sites respectively.

1. Create VxVM data volumes as backing store for the VxFS filesystem.

   [PrimarySite] # `vxassist -g dockerdg make vol1 1G`

2. Create and mount the filesystem.

   [PrimarySite] # `mkfs -t vxfs /dev/vx/rdsk/dockerdg/vol1`

   [PrimarySite] # `mkdir /vol1`

   [PrimarySite] # `mount -t vxfs /dev/vx/dsk/dockerdg/vol1 /vol1`

3. Create log volume (SRL) for VVR replication.

   [PrimarySite] # `vxassist -g dockerdg make srlvol 300m`

4. Repeat steps 1 and 3 on the secondary site.

5. Setup primary replication group (RVG).

   [PrimarySite] # `vradmin -g dockerdg createpri rvg vol1 srlvol`

6. Add secondary site.

   [PrimarySite] # `vradmin -g dockerdg addsec rvg <primarysite ip address> <secondarysite ip address>`

7. Start replication.

   [PrimarySite] # `vradmin -g dockerdg -a startrep rvg`

8. Create a Docker container.

   # `docker run -it --name container -v /vol1:/vol1 ubuntu /bin/bash`

To configure VVR and VFR for replication between DR sites:

See "Configure VVR and VFR in VCS GCO option for replication between DR sites" on page 160.

For information about configuring VVR/VFR-related resources, see the Storage Foundation and High Availability Solutions Replication Administrator's Guide.

For information about the VVR-related agents, see the Cluster Server Bundled Agents Reference Guide.

Limitations while managing Docker containers

- Administrative tasks: All VxFS and VxVM administrative commands, such as resize, add volumes, reorganize volume sets, so on are supported only on host nodes. These administrative commands cannot be executed inside Docker containers.

- Security-Enhanced Linux (SELinux): SELinux is a Linux kernel module that provides a mechanism for supporting access control security policies. For data volumes backed by VxFS mount points, SELinux needs to be in disabled or permissive mode on host nodes.
- Package installation only on host nodes: Installation and configuration of InfoScale solutions inside containers is not supported.
- Root volume: Veritas does not recommend exporting root volumes to Docker containers.
- Data loss because volume devices are not synchronized: If a volume is exported to a Docker container, some VxVM operations, such as removing volumes, deporting a disk group, renaming a volume, remirroring a disk group or volume, or restarting VxVM configuration daemon (vxconfigd) , can cause the volume device to go out of sync, which may cause data loss.

Section 5

Reference

Appendix A

Troubleshooting

This appendix includes the following topics:

Troubleshooting virtual machine live migration

A VCS cluster is formed between virtual machines (VMs) and one of the virtual machines is migrated from one host to another host. During a virtual machine migration, if the VM takes more than 16 seconds to migrate to the target node, one of the VMs panics. In this case, 16 seconds is the default value of the LLT `peerinact` parameter. You can increase the `peerinact` value to allow sufficient time for the VM to migrate. You can adjust this time based on the environment in which you initiate the VM migration.

To avoid false failovers for virtual machine migration, you can change the `peerinact` value using the following methods:

- Set the `peerinact` value dynamically using lltconfig command:

  ```
  # lltconfig -T peerinact:value
  ```

- Set the `peerinact` value in the `/etc/lltab` file to make the value persistent across reboots.

To set the peerinact value dynamically using lltconfig command

1. Determine how long the migrating node is unresponsive in your environment.
2. If that time is less than the default LLT peer inactive timeout of 16 seconds, VCS operates normally.

   If not, increase the peer inactive timeout to an appropriate value on all the nodes in the cluster before beginning the migration.

   For example, to set the LLT `peerinact` timeout to 20 seconds, use the following command:

   ```
   # lltconfig -T peerinact:2000
   ```

   The value of the `peerinact` command is in .01 seconds.

3 Verify that `peerinact` has been set to 20 seconds:

```
# lltconfig -T query

Current LLT timer values (.01 sec units):
 heartbeat    = 50
 heartbeatlo = 100
 peertrouble = 200
 peerinact    = 2000
 oos          = 10
 retrans      = 10
 service      = 100
 arp          = 30000
 arpreq          = 3000
 Current LLT flow control values (in packets):
 lowwater  = 40
```

4 Repeat steps 2 to 3 on other cluster nodes.

5 Reset the value back to the default `peerinact` value using the `lltconfig` command after the migration is complete.

To make the LLT peerinact value persistent across reboots:

- Append the following line at the end of `/etc/lltab` file to set the LT `peerinact` value to 20 seconds:

  ```
  set-timer peerinact:2000
  ```

  After appending the above line, `/etc/lltab` file should appear similar to the following:

  ```
  # cat /etc/lltab
  set-node sys1
  set-cluster 1234
  link eth2 eth-00:15:17:48:b5:80 - ether - -
  link eth3 eth-00:15:17:48:b5:81 - ether - -
  set-timer peerinact:2000
  ```

For more information on VCS commands, see the *Cluster Server Administrator's Guide*.

For attributes related to migration, see the *Cluster Server Bundled Agents Reference Guide*.

Live migration storage connectivity in a Red Hat Enterprise Virtualization (RHEV) environment

In a RHEV environment, if a virtual machine (VM) is migrating from one host to another and source host loses storage connectivity then the VM remains in the paused state. This issue is RHEV environment specific.

There is no workaround.

Troubleshooting Red Hat Enterprise Virtualization (RHEV) virtual machine disaster recovery (DR)

When you fail over the replication service group from one site to another, the hosts in the old site may go into the NON_RESPONSIVE state in the RHEV-M console.

To resolve the hosts in the NON_RESPONSIVE state in the RHEV-M console

1. Move the host into MAINTENENCE mode.
2. Try to ACTIVATE the host using the RHEV-M console.
3. If the issue still persists, contact Redhat Support to get it resolved.

The KVMGuest resource may remain in the online state even if storage connectivity to the host is lost

When a virtual machine is running on a physical host and loses storage connectivity, the virtual machine goes into the PAUSED state. However, the virtual machine process is still running. The KVMGuest resource monitoring the virtual machine reports the state as ONLINE as the virtual machine process is still running and no failover is initiated. The KVMGuest resource is not aware of the storage situation, and therefore does not take any action.

If this issue occurs, either offline the service group or manually switch the service group. This shuts down the virtual machine and starts the virtual machine on another node.

VCS initiates a virtual machine failover if a host on which a virtual machine is running loses network connectivity

When a virtual machine is running on a physical host and loses network connectivity, such as a public or private communication channel, VCS on each node is not able to communicate. This is a classic split brain situation. VCS running on a node thinks that the other node has crashed and initiates a virtual machine failover.However, the virtual machine is still running on one node while VCS attempts to start same virtual machine on another node.

If this issue occurs, configure disk based fencing to prevent a split brain situation due to a network partition.

Virtual machine start fails due to having the wrong boot order in RHEV environments

When creating a virtual machine, you can specify the boot order. If a virtual machine has the following boot order, the virtual machine start fails as it is not able to find the CD-ROM:

- CD-ROM
- Hard disk

If VCS initiated the virtual machine start, any associated KVMGuest resources also fail. This issue is due to RHEV behavior.

If this issue occurs, manually edit the boot order and remove the CD-ROM from the boot sequence. Then re-initiate the virtual machine start using VCS or the RHEV-M console.

Virtual machine hangs in the wait_for_launch state and fails to start in RHEV environments

When a virtual machine start is initiated through the RHEV-M console, the virtual machine may hang in the wait_for_launch state, and then fails to start. This issue occurs when the `libvirtd` service is unable to process the virtual machine start operation.

There is no workaround.

VCS fails to start a virtual machine on a host in another RHEV cluster if the DROpts attribute is not set

In the RHEV environment, every host is part of a RHEV cluster. In a local high availability scenario, hosts forming a VCS cluster should be part of a single RHEV cluster. However, in disaster recovery scenarios, you can configure all hosts on the primary site in one RHEV cluster and all hosts on the secondary site in a different RHEV cluster, though they are all part of the same datacenter. During a site failover, when the `DROpts` attribute is set, VCS changes the virtual machine host as per the new RHEV cluster.

If the `DROpts` attribute is not set, VCS does not allow a host from a different RHEV cluster to start the virtual machine. This issue occurs because virtual machine migration does not work across RHEV clusters. Therefore, VCS fails to start the virtual machine on a host that is part of a different cluster.

Veritas recommends configuring hosts in different clusters only in a disaster recovery configuration, and setting the `DROpts` attribute of the KVMGuest agent. For a local high availability scenario, you do not need to set the `DROpts` attribute, and all the hosts forming a VCS cluster should be part of the same RHEV cluster.

Virtual machine fails to detect attached network cards in RHEV environments

A virtual machine may fail to detect an attached network interface. This issue is due to RHEV behavior.

There is no workaround.

The KVMGuest agent behavior is undefined if any key of the RHEVMInfo attribute is updated using the -add or -delete options of the hares -modify command

If you modify any key of the `RHEVMInfo` attribute using the `-add` or `-delete` options of the `hares -modify` command, the `RHEVMInfo` attribute information sequence changes and can cause the KVMGuest resource behavior to be undefined. The `-add` option adds a new key to any attribute, and the `-delete` option deletes a key

from any attribute. These two options should not be used to configure the `RHEVMInfo` attribute.

Use the `-update` option of the `hares -modify` command to modify attribute keys:

```
# hares -modify resource_name RHEVMInfo -update key_name value
```

For example:

```
# hares -modify vmres RHEVMInfo -update User "admin@internal"
```

RHEV environment: If a node on which the VM is running panics or is forcefully shutdown, VCS is unable to start the VM on another node

In a RHEV environment, if a node on which a virtual machine is running panics or is forcefully shutdown, the state of that virtual machine is not cleared. RHEV-M sets the VM to UNKNOWN state and VCS is unable to start this virtual machine on another node. You must initiate manual fencing in RHEV-M to clear the state.

This is not a VCS limitation because it is related to RHEV-M design. For more information, refer *Red Hat Enterprise Virtualization 3.4 Technical Guide*.

To initiate manual fencing in RHEV-M and clearing the VM state

1. In the RHEVMinfo attribute, set the UseManualRHEVMFencing key to 1.

   ```
   UseManualRHEVMFencing = 1
   ```

2. Override the resource attribute:

   ```
   hares -override resource_name OnlineRetryLimit
   ```

3. Modify the OnlineRetryLimit attribute value to 2:

   ```
   hares -modify resource_name OnlineRetryLimit 2
   ```

After you clear the state of the VM, VCS starts the VM on another node.

The following is a sample resource configuration of RHEV-based disaster recovery:

```
group rhev_sg (
SystemList = { rhelh_a1 = 0, rhelh_a2 = 1 }
TriggerPath ="bin/triggers/RHEVDR"
PreOnline=1
OnlineRetryLimit = 1
```

```
)
KVMGuest rhev_fo (
RHEVMInfo = { Enabled = 1, URL =
"https://192.168.72.11:443",
User = "admin@internal",
Password = flgLglGlgLglG,
Cluster = RHEV-PRIM-CLUS,
UseManualRHEVMFencing = 1 }
GuestName = swvm02
OnlineRetryLimit = 2
)
// resource dependency tree
//
// group rhev_sg
// {
// KVMGuest rhev_fo
// }
```

Appendix B

Sample configurations

This appendix includes the following topics:

Sample configuration in a KVM environment

You can use any of the followoing sample confirgurations:

- Sample configuration 1: Native LVM volumes are used to store the guest image
- Sample configuration 2: VxVM volumes are used to store the guest image
- Sample configuration 3: CVM-CFS is used to store the guest image

Sample configuration 1: Native LVM volumes are used to store the guest image

```
group kvmtest1 (
SystemList = { sys1 = 0, sys2 = 1 }
)
KVMGuest res1 (
GuestName = kvmguest1
GuestConfigFilePath = "/kvmguest/kvmguest1.xml"
DelayAfterGuestOnline = 10
DelayAfterGuestOffline = 35
)
Mount mnt1 (
BlockDevice = "/dev/mapper/kvmvg-kvmvol"
MountPoint = "/kvmguest"
```

```
FSType = ext3
FsckOpt = "-y"
MountOpt = "rw"
)
LVMLogicalVolume lv1 (
VolumeGroup = kvmvg
LogicalVolume = kvmvol
)
LVMVolumeGroup vg1 (
VolumeGroup = kvmvg
)
res1 requires mnt1
mnt1 requires lv1
lv1 requires vg1
```

Sample configuration 2: VxVM volumes are used to store the guest image

```
group kvmtest2 (
SystemList = { sys1 = 0, sys2 = 1 }
)
KVMGuest res1 (
GuestName = kvmguest1
GuestConfigFilePath = "/kvmguest/kvmguest1.xml"
DelayAfterGuestOnline = 10
DelayAfterGuestOffline = 35
)
Mount mnt1 (
BlockDevice = "/dev/vx/dsk/kvmvg/kvmvol"
MountPoint = "/kvmguest"
FSType = vxfs
FsckOpt = "-y"
MountOpt = "rw"
)
Volume vol1 (
Volume = kvm_vol
DiskGroup = kvm_dg
)
DiskGroup dg1 (
DiskGroup = kvm_dg
)
res1 requires mnt1
```

```
mnt1 requires vol1
vol1 requires dg1
```

Sample configuration 3: CVM-CFS is used to store the guest image

```
group kvmgrp (
SystemList = { kvmpm1 = 0, kvmpm2 = 1 }
)
KVMGuest kvmres (
GuestName = kvmguest1
GuestConfigFilePath = "/cfsmount/kvmguest1.xml"
DelayAfterGuestOnline = 10
DelayAfterGuestOffline = 35
)

kvmgrp requires group cvm online local firm

group cvm (
SystemList = { kvmpm1 = 0, kvmpm2 = 1 }
AutoFailOver = 0
Parallel = 1
AutoStartList = { kvmpm1, kvmpm2 }
)
CFSMount cfsmount (
MountPoint = "/cfsmount"
BlockDevice = "/dev/vx/dsk/cfsdg/cfsvol"
)
CFSfsckd vxfsckd (
)
CVMCluster cvm_clus (
CVMClustName = kvmcfs
CVMNodeId = { kvmpm1 = 0, kvmpm2 = 1 }
CVMTransport = gab
CVMTimeout = 200
)
CVMVolDg cfsdg (
CVMDiskGroup = cfsdg
CVMVolume = { cfsvol }
CVMActivation = sw
)
CVMVxconfigd cvm_vxconfigd (
Critical = 0
CVMVxconfigdArgs = { syslog }
```

```
)

cfsmount requires cfsdg
cfsmount requires cvm_clus
cvm_clus requires cvm_vxconfigd
vxfsckd requires cvm_clus
```

Sample configurations for a Red Hat Enterprise Virtualization (RHEV) environment

Sample configuration for a RHEV-based service group:

```
group rhev_grp1 (
        SystemList = { sys1 = 0, sys2 = 1 }
        )

        KVMGuest kvmres1 (
                RHEVMInfo = { Enabled = 1,
                         URL = "https://rhevm-server.example.com:443",
                         User = "admin@internal"
                         Password = bncNfnOnkNphChdHe,
                         Cluster = dc2_cluster1,
                         UseManualRHEVMFencing=1 }
                GuestName = rhevvm1
                DelayAfterGuestOnline = 20
                DelayAfterGuestOffline = 35
                )
```

Sample configuration for an AD-based domain:

```
include "types.cf"

cluster kvmtest (
        UserNames = { admin = bQRjQLqNRmRRpZRlQO }
        Administrators = { admin }
        )

system sys1 (
        )

system sys2 (
        )
```

```
group virt_grp (
        SystemList = { sys1 = 0, sys2 = 1 }
        )

        KVMGuest virt_res (
                RHEVMInfo = { Enabled = 1,
                        URL = "https://rhevm.example.com:443",
                        User = rhevmadmin@example.com",
                        Password = codOgoPolOqiDieIf,
                        Cluster = cluster_NFS,
                        UseManualRHEVMFencing=0 }
                GuestName = VM1
                )
```

Sample configuration for a RHEV-based disaster recovery service group:

```
group VM_SG (
        SystemList = { rhelh_a1 = 0, rhelh_a2 = 1 }
        TriggerPath = "bin/triggers/RHEVDR"
        PreOnline = 1
        OnlineRetryLimit = 2
        )

        KVMGuest kvm_res (
                RHEVMInfo = { Enabled = 1, URL = "https://192.168.72.11:443",
                        User = "admin@internal",
                        Password = CQIoFQf,
                        Cluster = RHEV-PRIM-CLUS,
                        UseManualRHEVMFencing = 1 }
                GuestName = swvm02
                DROpts = { ConfigureNetwork = 1,
                        IPAddress = "192.168.74.21",
                        Netmask = "255.255.252.0",
                        Gateway = "192.168.74.1",
                        DNSServers = "143.127.176.14",
                        DNSSearchPath = "rhevdc.com",
                        Device = eth0 }
                )

        requires group STORAGE online global soft


        // resource dependency tree
        //
```

```
        //      group VM_SG
        //      {
        //      KVMGuest kvm_res
        //      }


group STORAGE (
        SystemList = { rhelh_a1 = 0, rhelh_a2 = 1 }
        ClusterList = { RHEV_SEC = 0, RHEV_PRIM = 1 }
        TriggerPath = "bin/triggers/RHEVDR"
        TriggersEnabled = { POSTONLINE }
        )

        SRDF srdf_res1 (
                GrpName = rhevdr
                )

        SRDF srdf_res2 (
                GrpName = rhevdr2
                )



        // resource dependency tree
        //
        //      group STORAGE
        //      {
        //      SRDF srdf_res1
        //      SRDF srdf_res2
        //      }
```

Sample configuration for a multi-resource configuration in a RHEV environment:

```
system sys1 (
        )

system sys2 (
        )

group rhevgrp1 (
        SystemList = { sys1 = 0, sys2 = 1 }
        )

        KVMGuest vmres1 (
```

```
                RHEVMInfo = { Enabled = 1,
                        URL = "https://rhevm.example.com:443",
                        User = "admin@internal",
                        Password = FRGrJRsROrTLgLHlI,
                        Cluster = vcs_clus,
                        UseManualRHEVMFencing = 0 }
                GuestName = vcsvm1
                DelayAfterGuestOnline = 10
                DelayAfterGuestOffline = 35
                )


group rhevgrp2 (
        SystemList = { sys1 = 0, sys2 = 1 }
        )

        KVMGuest vmres2 (
                RHEVMInfo = { Enabled = 1,
                        URL = "https://rhevm.example.com:443",
                        User = "admin@internal",
                        Password = FRGrJRsROrTLgLHlI,
                        Cluster = vcs_clus,
                        UseManualRHEVMFencing = 0 }
                GuestName = vcsvm2
                DelayAfterGuestOnline = 7
                DelayAfterGuestOffline = 30
                )
```

Sample configuration for RHEV virtual machine migration:

```
group rhevgrp (
        SystemList = { sys1 = 0, sys2 = 1 }
        )

        KVMGuest rhevres (
                RHEVMInfo = { Enabled = 1,
                        URL = "https://rhevm.example.com:443",
                        User = "admin@internal",
                        Password = AMBmEMnMJmOGbGCgD,
                        Cluster = rhev_cluster,
                        UseManualRHEVMFencing=1 }
                GuestName = rhevvm1
                DelayAfterGuestOnline = 15
```

```
DelayAfterGuestOffline = 45
)
```

Appendix C

Where to find more information

This appendix includes the following topics:

Veritas InfoScale documentation

The latest documentation is available on the Veritas Services and Operations Readiness Tools (SORT) website in the Adobe Portable Document Format (PDF).

See the release notes for information on documentation changes in this release.

Make sure that you are using the current version of documentation. The document version appears on page 2 of each guide. The publication date appears on the title page of each document. The documents are updated periodically for errors or corrections.

https://sort.veritas.com/documents

You need to specify the product and the platform and apply other filters for finding the appropriate document.

Note: The commands used for the Red Hat Enterprise Linux (RHEL) operating system in the Veritas InfoScale documents also apply to supported RHEL-compatible distributions.

Linux virtualization documentation

For Red Hat documentation:

- Red Hat Enterprise Linux (RHEL):
  https://access.redhat.com/site/documentation/Red_Hat_Enterprise_Linux/
- Red Hat Enterprise Virtualization (RHEV):
  https://access.redhat.com/site/documentation/Red_Hat_Enterprise_Virtualization/
- KVM Whitepaper:
  http://www.redhat.com/resourcelibrary/whitepapers/doc-kvm
- KVM Open source Project Site:
  http://www.linux-kvm.org/page/Main_Page

For SUSE:

- SUSE Linux Enterprise Server (SLES):
  http://www.suse.com/documentation/sles11/book_kvm/?page=/documentation/sles11/book_kvm/data/book_kvm.html
- For SLES11SP3 installation information:
  http://www.suse.com/documentation/sles11
  For a full set of features and capabilities, see the SUSE documentation.

Service and support

To access the self-service knowledge base, go to the following URL:

https://www.veritas.com/support/en_US.html

About Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a Web site that automates and simplifies some of the most time-consuming administrative tasks. SORT helps you manage your datacenter more efficiently and get the most out of your Veritas products.

SORT can help you do the following:

Prepare for your next installation or upgrade	■ List product installation and upgrade requirements, including operating system versions, memory, disk space, and architecture. ■ Analyze systems to determine if they are ready to install or upgrade Veritas products. ■ Download the latest patches, documentation, and high availability agents from a central repository. ■ Access up-to-date compatibility lists for hardware, software, databases, and operating systems.
Manage risks	■ Get automatic email notifications about changes to patches, array-specific modules (ASLs/APMs/DDIs/DDLs), and high availability agents from a central repository. ■ Identify and mitigate system and environmental risks. ■ Display descriptions and solutions for hundreds of Veritas error codes.
Improve efficiency	■ Find and download patches based on product version and platform. ■ List installed Veritas products and license keys. ■ Tune and optimize your environment.

Note: Certain features of SORT are not available for all products. Access to SORT is available at no extra cost.

To access SORT, go to:

https://sort.veritas.com